applied sciences

MDPI

Article

# Mini-MES: A Microservices-Based Apps System for Data Interconnecting and Production Controlling in Decentralized Manufacturing

**Pulin Li, Pingyu Jiang * and Jiajun Liu**

State Key Laboratory for Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an 710049, China
* Correspondence: pjiang@mail.xjtu.edu.cn

Received: 19 July 2019; Accepted: 26 August 2019; Published: 5 September 2019


**Abstract:** Manufacturing factories and their resources are moving towards being networked, flat, flexible, and diversified in the context of social manufacturing. It is urgent to manage the increasingly complex and variable manufacturing processes effectively in such cross-factory production. This paper proposes a brand-new microservices-based Mini-MES ($M^2ES$) Apps System, which adopts the decentralized and distributed architecture to comply with the needs of software supported manufacturing. The basic properties of the $M^2ES$ Apps System, such as its running logic and its data organization, are discussed, and a real-world industrial case is described to illustrate how it works. Results show that the proposed $M^2ES$ Apps System has better performance, reliability, and scalability than the initial information systems.

**Keywords:** manufacturing execution systems; MES; industrial informatics; mini program; microservices; decentralized manufacturing; industrial dataspace; SOA

## 1. Introduction

Manufacturing Execution Systems (MES), a kind of computerized system, is widely used in manufacturing, to track and document the transformation of raw materials into finished goods. MES has become involved in production planning and control, and it significantly evolved into a more and more powerful system as computing technologies and integration levels become more advanced [1]. MES, together with other information systems, such as Customer Relationship Management (CRM), Enterprise Resource Planning (ERP), Supply Chain Management (SCM), and Product Data Management (PDM), is widely employed by manufacturers to deal with production management and control [2].

Many manufacturers looking to implement a new MES system will, no doubt, end up considering the industry's some biggest behemoths: SAP [3], Oracle [4], and Siemens [5]. These vendors are clear market share leaders and have well-established product lines. However, all the above MES systems are developed with a centralized architecture, where various data, resources, and functions of its modules are highly coupled with each other. What's more, the data type and processing flow should be figured out and defined before starting programming due to the complexity of the data and business logic [6]. This situation could lead to the development of an oligopoly where once a manufacturer opts for one system, there is almost no opportunity to use others because of the higher replacement cost.

Nowadays, manufacturers must respond to local and global market shifts as well as to changing regulatory and customer demands more quickly than their competitors. This means manufacturers have to face the music when upgrading their MESs to adapt to new production needs. Their systems' vendors always tend to quote at an astronomical price, since only they can access their own software to customize it. Vendors like SAP have already taken the upgradation and customization issues into consideration, and some Application Program Interfaces (APIs) are released along with the

standardized products. But this still brings about some difficulties in customized programming and personalized maintenance in a non-open source system.

In the meantime, several strategies for manufacturing have been developed, such as Industry 4.0 in Germany, Industrial Internet in the USA, and Made in China 2025 in China [7]. These initiatives encourage the applications of New Generation of Information Technologies (NGITs) [8] in manufacturing, such as big data, edge computing, artificial intelligence, and so on. NGITs drive manufacturing to become more and more flat and interconnected. More participants and roles are involved in the same production process than ever, which makes the interconnection of software and sharing of data more and more important. In terms of MES, its goal has become collaborative control all over the industry chain instead of accomplishing the internal planning or scheduling inside an enterprise or a workshop. Objectively, these changes have driven the evolution of data governance from centralized to distributed and decentralized.

It is believed that the MES in the future will take over the manufacturing ecosystem, which is not limited to its original definition. It will combine with the information systems of ERP, CRM, and others, to extend more customized functions. Also, a future MES must have the microservice-based feature to fit in the dispersion of data and use the lightweight and flexible architecture to adapt to the more complex functions in discrete manufacturing. In other words, embedded Apps in the future will not need to be downloaded and installed [9]. Users will only need to Open, Use, Share, and Close. As a consequence, the characteristics of next-generation MES can be summarized as follows.

### *1.1. Supporting the Distributed and Decentralized Manufacturing*

The boundaries between different workshops, different factories, and different enterprises have been blurred in the context of Service-based Manufacturing and Cloud Manufacturing. Software interoperability and data sharing have become major trends in current manufacturing. While existing MES products have achieved great success in centralized manufacturing [10], there are some difficulties in supporting life-cycle manufacturing at the industrial ecosystem level. Lots of suppliers, outsourcers, vendors, logistics providers, and even the customers prefer to access the information system by themselves to track or control the production. So, it seems possible to realize distributed and decentralized manufacturing supported with software by considering the security and permissions issues [11]. With this architecture and edge computing technology, Apps can be deployed anywhere in the Industrial Internet of Things, further enabling an on-demand configuration.

### *1.2. Enabling Extensible Ongoing Development*

Functional modules are relatively fixed in existing MES systems, which means they can only use the established features to help to control the production. Nevertheless, business logic in the era of intelligent manufacturing or social manufacturing will change frequently. It is important to design a specific new environment for service-oriented App to enable extensible ongoing development. In the new development environment, the addition of a new App will definitively not affect all previous ones, and the new App can request the raw or processed data from other existing Apps. When a new production scenario develops that requires the supporting of software, the only thing to do is to customize a functionally independent App for this new scenario, and the rest of the system will continue operating without any interruption. Besides this, an efficient workshop dataspace [12] will be also employed to manage the heterogeneous and heterologous data [13].

### *1.3. Lightweighting Application Logic*

From the perspective of the entire industrial ecosystem, the future MES will not only consider the needs of deployment and application for core enterprises, but will also think about small and micro individuals. And these small and micro enterprises will not frequently invest in high-configured servers and clients. They just want to make full use of their limited hardware and software resources to develop an easy-to-use App with a simple function. In recent years, lightweight architecture has been

becoming more and more popular. Some HTML5 Web-App based software architectures have begun to emerge [14], such as the WeChat Mini Programs [15]. As part of the next generation of the mobile Internet, Web-App and Mini Programs are developing rapidly toward the "micro, light, and small", and developers may develop a new App or Program quickly. In addition, these Apps and Programs may be easily accessed and easily disseminated in Platform [16].

### 1.4. Achieving Industry 4.0-Oriented Intercommunication

In a smart factory for Industry 4.0, smart machines will deal with the production orders, which are composed of the smart workpieces, via the Human-Machine Interface (HMI) or the social sensors [17,18]. Under these circumstances, smart devices will get the same positions and permissions as ordinary workers, and the users of MES will no longer limited be to real people, but will extend to machines, tools, and workpieces. Intercommunication in this manufacturing scenario would be more like a Multi-Agent System (MAS) [19], where the interactions between any two Agents are Peer-to-Peer, instead of using the traditional industrial Fieldbus.

### 1.5. Ensuring High-Level Safety and Reliability

The production logic is becoming more sophisticated in intelligent manufacturing, and the data types and data dimensions are becoming more disordered. This puts high demands on the authority and data security management in the next generation MES. It is necessary to ensure efficient and reliable running of the system, and to prevent sensitive data from being leaked and tampered with in a more open-up manufacturing ecosystem than ever.

For a better application of MES in the context of intelligent manufacturing, we propose a brand-new microservices-based Mini-MES ($M^2$ES) Apps System. It can be seen as the next generation MES, which is developed for the software supporting in more complex production situations. $M^2$ES Apps System is defined as a computerized system with microservices-based architecture, including a series of feature-independent Apps, to help to cope with any life-cycle production issues. $M^2$ES Apps System is also a highly scalable system, and each App in it has a lightweight architecture. $M^2$ES Apps can work independently and be networked via the Industrial Internet. Also, some Apps can be combined in order to solve some complicated cases. Additionally, since the manufacturing process has many similar business scenarios, the $M^2$ES Apps System can meet the requirements of reusing in analogical scenes at different time, and the Apps are easy to migrate and to promote.

As an information system adapts to new manufacturing paradigms and new production models, $M^2$ES Apps System will enable manufacturing processes to receive more support from computer software with less effort required on development. This paper focuses on some basic properties, especially its data organization and dataspace, of $M^2$ES Apps System and its applications. The rest of the paper will be arranged as follows: Section 2 reviews the related work on the Distributed Information System, Microservices-based Architecture, and Industrial Dataspace. Section 3 clarifies the running logic of the $M^2$ES Apps System and describes its framework of data organization, together with some key enabling technologies. Furthermore, an industrial case is described in Section 4 to illustrate the application of the $M^2$ES Apps System. Finally, discussions and conclusions are given.

## 2. Related Work

Several computer-aided methods and NGIT have been developed for manufacture controlling in some new production scenarios over the years. This section will elaborate on the related works from the following three aspects.

### 2.1. Distributed Information System

Manufacturing ecosystem for individualized products in the context of advanced manufacturing paradigms, especially in Cloud Manufacturing [20] and Social Manufacturing [21], is now likely to involve a number of distributed autonomous organizations. In fact, quite a bit of research [22] has

been done on logical decentralization of the information system, and MAS can be seen as a typical kind of physical decentralization inside a smart factory [23]. MAS has succeeded in superseding traditional MES functions through distributed autonomy of numerous resources [24]. The autonomous communication and negotiation in distributed manufacturing can be represented as individual behaviors among the agents. Although the centralization versus decentralization debate has been with us since early computing, there are no good or bad conclusions [25]. Some scholars also tried to use the Ontology [26] or API [27] to realize the interconnection of existing information systems in the workshop floor. In addition, an information system orienting Industry 4.0 should sustain multi-enterprise collaboration [28], which requires MES to have the function of authority controlling when running cross-enterprise or cross-workshop.

### 2.2. Microservices-Based Architecture

Microservices-based architecture is an architectural style that structures an App as a collection of services that are highly maintainable and testable, loosely coupled, independently deployable, organized around business capabilities [29]. As a popular distributed App development ecosystem, Microservices-based architecture enables rapid authorization and management [30]. Many famous and large Internet companies have provided authorization services, such as Google Accounts, Facebook, Alipay, WeChat, Microblogging, QQ, and other third-party interactive tools [31]. These architectures offer a lightweight infrastructure that allows developers to pay more attention to the core business workflow design and help users enjoy the process of Opening, Using, Sharing, and Closing. At the moment, microservices-based architecture is still at an early age in the field of manufacturing or production, and there are only some conceptual proposals about microservices either in the digital factory [32] or in IoT-based manufacturing [33] at present.

### 2.3. Industrial Dataspace

Dataspaces was first introduced as a novel abstraction for data management [34] in 2005. In the informatics field, modeling of data, integrating and relationship discovering of entities, indexing of information, etc. have been studied [35]. In the manufacturing field, industrial dataspace was regarded as a broker to run Cyber-Physical Systems (CPS) [36] by mediating between bottom data and upper Apps via mappings. And it refers to the semantic representation of multi-source and heterogeneous data [37]. As to industrial applications, Ontology-Based Data Access (OBDA) and Knowledge Graph (KG) were widely used to integrate, store, index, semantically query, and knowledge reason [38]. General Electric (GE) also proposed a semantic-driven framework *SemTK*, which realized storage and access of heterogeneous big data. In addition, the benefits of constructing and querying *Polystore* knowledge graphs with *SemTK* were employed in four industrial cases [39].

## 3. Running Logic and Data Organization of the M$^2$ES Apps System

In order to design a lightweight and distributed architecture of M$^2$ES Apps System, which is very different from the traditional MES, this section elaborates its software implementation architecture and data organization logic, as shown in Figure 1.

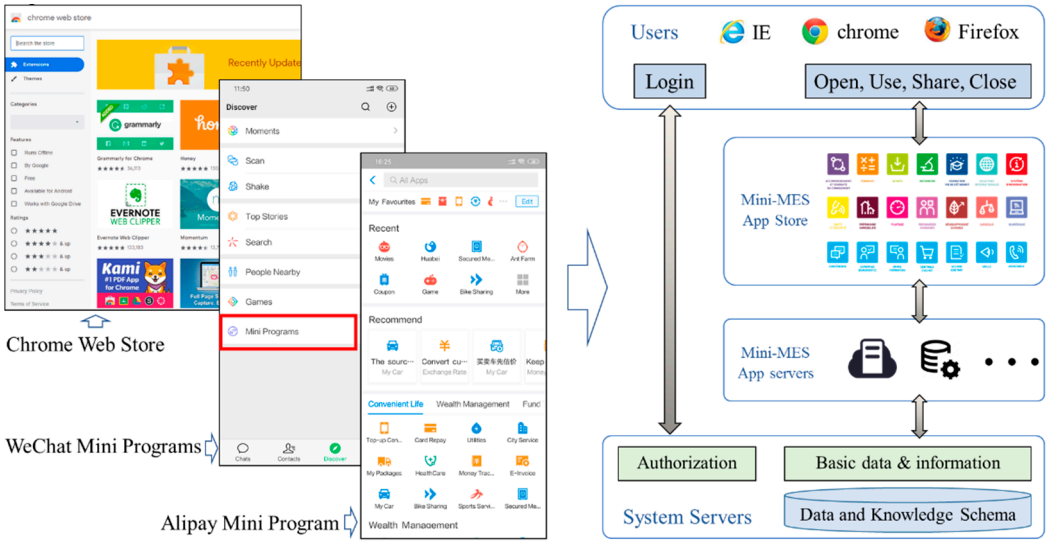



**Figure 1.** The architecture of the M$^2$ES Apps System.

### *3.1. Characteristics of M$^2$ES Apps System*

#### 3.1.1. M$^2$ES Apps System Architecture

Manufacturing processes involve a variety of physical resources and operational processes, so the first step in establishing M$^2$ES Apps System is defining all the related resources and processes uniquely. The concept of industrial dataspace is introduced to meet requirements imposed by M$^2$ES Apps System, which refers to the semantic representation [40] of multi-source and heterogeneous data. Figure 2 shows how this M$^2$ES Apps system works.

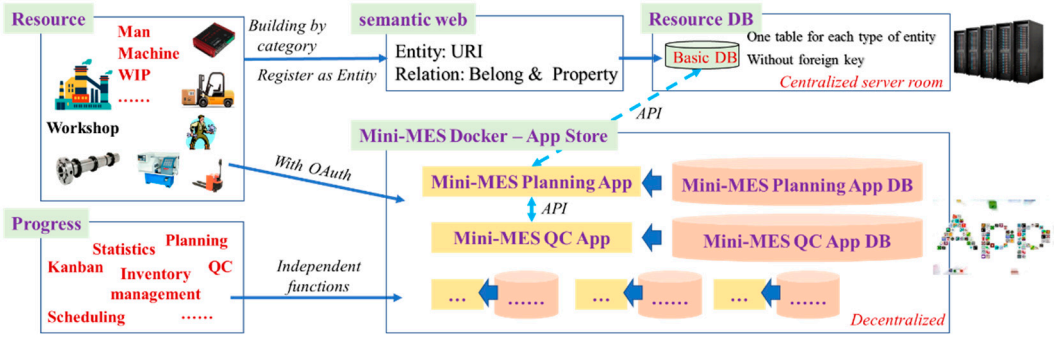



**Figure 2.** How the M$^2$ES Apps System works.

Each resource, whether it is a physical one like a machine or a virtual one like an order, will be registered as an entity in industrial dataspace. Then the central Resource DB, which is also called Basic DB, assigns a data table for each of them. This table will save the properties and status data of the entity. The Resource DB will be placed in the centralized computer room normally, and only data that is not relevant to the manufacturing process will be saved in.

As for the processing and implementation of the function, they are handled by each distributed M$^2$ES App. Every App is developed, debugged, and deployed independently under a unified instruction, and each of them has its own separate database. The functionality of an App is relatively simple and independent strictly, and resource entities need to follow the *OAuth 2.0* [41] architecture in logging in and using. What's more, M$^2$ES Apps can also request and pull data from others through an Application Programming Interface (API). It should be noted that module calls in cross-App are not permitted, and only API-based data requests can be allowed.

### 3.1.2. Configuration and Running of M²ES Apps

Running an M²ES App can be divided into four major stages mainly, seen in Figure 3.

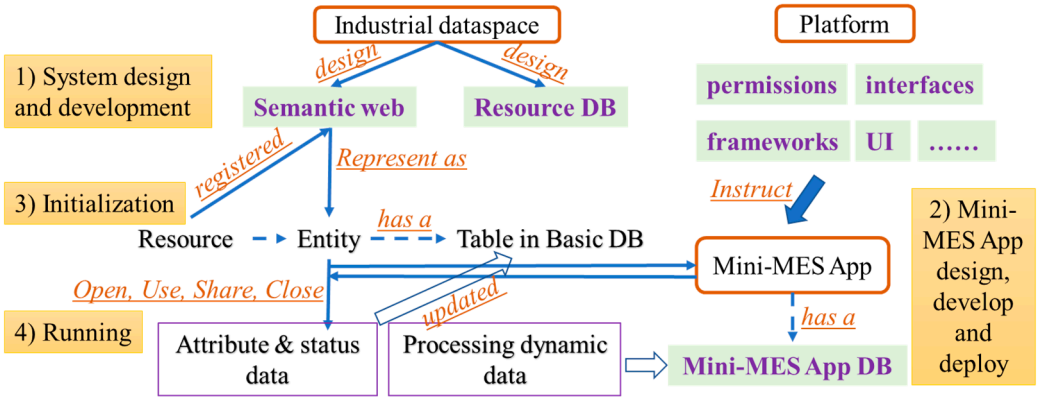



**Figure 3.** Running logic of the M²ES Apps.

(1) System design and development. It can be subdivided into two segments. The first one is designing a description framework of resources, that is, the industrial dataspace; the other is building the software platform with permissions, authorization, API, frameworks, UI, and other specifications.

(2) M²ES App design, develop and deploy. The M²ES App is designed and developed according to the relevant functions. Each App will get an isolated DB that stores process-related data only, by using the protocol defined in stage (1).

(3) Initialization. All resources are registered as entities in the industrial dataspace, and each of them gets its own data table in the basic DB. Users, represented as entities, can log in the M²ES App with its Uniform Resource Identifier (URI) [42].

(4) Running. Entities (users) will Open, Use, Share, and Close the Apps. The change of the attribute & status data of the entity itself is updated to the basic DB. And dynamic data generated in the course of the operation will be saved in the M²ES App DB. Obviously, this stage is the core of the M²ES Apps System, so its workflow will be elaborated on in detail.

### 3.1.3. Data Organizational Logic When Running an M²ES App

As mentioned above, each M²ES App performs only one independent function. What it needs to do is to process the input data, calculate and optimize, and format the output in accordance with certain organizational logics. Figure 4 shows how a user (entity) uses an M²ES App.

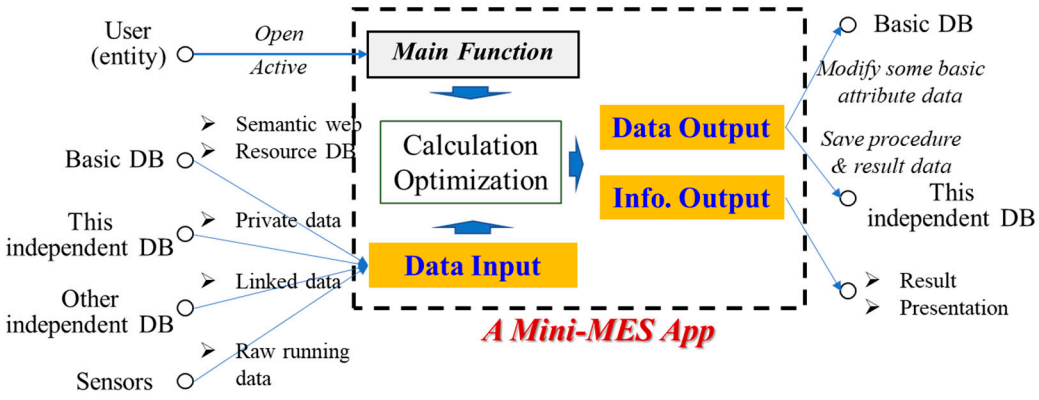



**Figure 4.** Input & Output of an M²ES App.

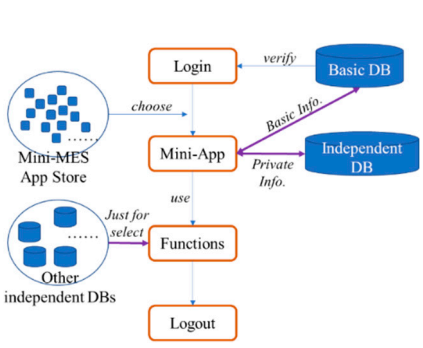
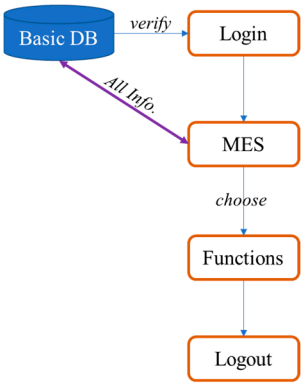
When the user opts an M$^2$ES App from the App Store, this selected will collect data as needed. The input data of the App is primarily from four aspects. The first is the attributes and the status data of resource entity from the Basic DB. The second is the private data taken out from the independent DB of this App according to the entity's URI. The third is the linked data from other Apps' independent DBs, which is pulled according to the entity's URI via API. The last is the raw running data from sensors, which represents the working conditions of equipment and devices.

The output of an M$^2$ES App usually includes two aspects, i.e., the running results and newly generated data. Running results are used to guide production and decision making, and all procedures and final data will be saved reliably. Some basic attribute data will be modified in Basic DB, and procedure and result data related to the function of the App will be stored in its independent DB.

#### 3.1.4. Comparative Analysis with Traditional MES

As M$^2$ES Apps can be defined as the next generation MES, it is necessary to figure out the differences between M$^2$ES Apps System and the traditional MES clearly, as shown in Table 1.

**Table 1.** Differences between M$^2$ES Apps System and Traditional MES.

| | | **M$^2$ES Apps System** | **Traditional MES** |
|---|---|---|---|
| Architectures | Computing | Distributed edge computing | Centralized computing |
| | Database | Independent DB for each App | Unified DB |
| | Functions | Functional decoupling | Functional correlative |
| Development | Development logic | Only considering core scenarios | Considering all possible scenarios at the beginning |
| | Database design | Design while developing | Design the database at the beginning |
| | Function adding | Develop a new App to implement new functions | Redesign the database; Add new functions considering conflicts with existing functions |
| | Function modifying | Modify an App's functions | Redesign the database; Consider conflicts with new functions |
| | Function deleting | Removed from the App Stores | Redesign the database; Consider conflicts with new functions |
| Usability | Logic |   |   |
| | Authority | Different users with different Apps | Different users with different functions |
| | Running data | Independent storage | Unified storage |

### 3.2. *Identification of Key Enabling Technologies for M$^2$ES Apps System*

Developing M$^2$ES Apps System needs the support of some key enabling technologies. Therefore, it is quite important to identify what key enabled technologies for M$^2$ES Apps System are. In general,

there are five main supporting technologies to ensure the smooth operation of M$^2$ES Apps System, which are as follows.

#### 3.2.1. Creating Data and Knowledge Graph

As mentioned in Section 3.1.1, the Semantic Web seems to be the basis of M$^2$ES Apps System. So, a schema for entity-centered resources will be created in order to describe the resources and manage them. Semantic technologies, such as especially OWL 2 ontology, are ideal models for the heterogeneous resources to enable the data exchange across different entities [43]. OWL 2 provides a rich and flexible modeling language for creating a domain ontology, which uses shared vocabularies to describe classes, properties, individuals, and data values. An ontology model of an entity can be denoted as

$$O = \{URI, C, P\} \tag{1}$$

where $C$ denotes the ontology classes and class hierarchy. $P$ stands for the property collection of the entity.

In the meantime, it's needed to create a data Table in the Basic DB for each entity resource $O$ in the operating level, which can save the properties and status data of the registered resource in the form of "*propertyName: propertyValue*".

#### 3.2.2. Apply for Webpage Authorization for an M$^2$ES App

Should the user visit an M$^2$ES App, the App can obtain basic information on users via the authorization mechanism, thereby achieving business logic. Specifically, there are four steps in authorizing an App, which are shown in Figure 5.

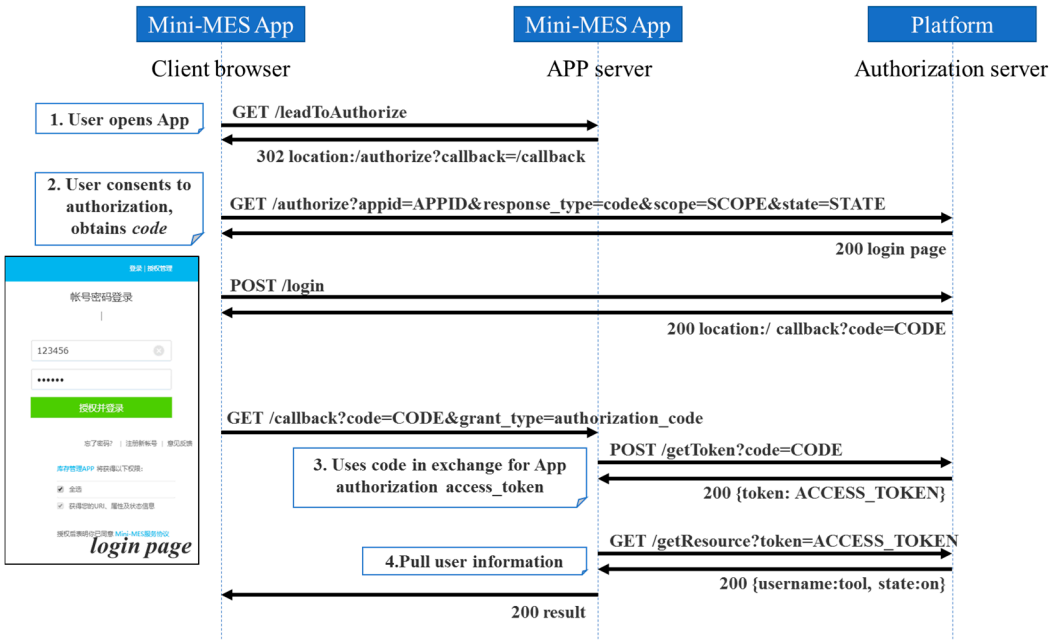



**Figure 5.** App authorization supported by OAuth2.0.

The *code* acts as a note for exchanging *access_token*, with the *code* used for webpage authorization being different each time. Each *code* can be only used once and will expire automatically in 5 min.

#### 3.2.3. Process Functional Decomposition and its App's Design

MES is intended to track and document the transformation of raw materials to finished goods frequently, which involves very numerous and interrelated processes. In order to ensure the

independence of each App's functions, it is necessary to model the processes and decompose the functions of the whole manufacturing task. Finer granularity always goes with more Apps, which will decrease convenience. So, it is vital to decompose the functions with appropriate granularity. Figure 6 shows a kind of typical decomposition of manufacturing tasks, in which each function corresponds to an M$^2$ES App.

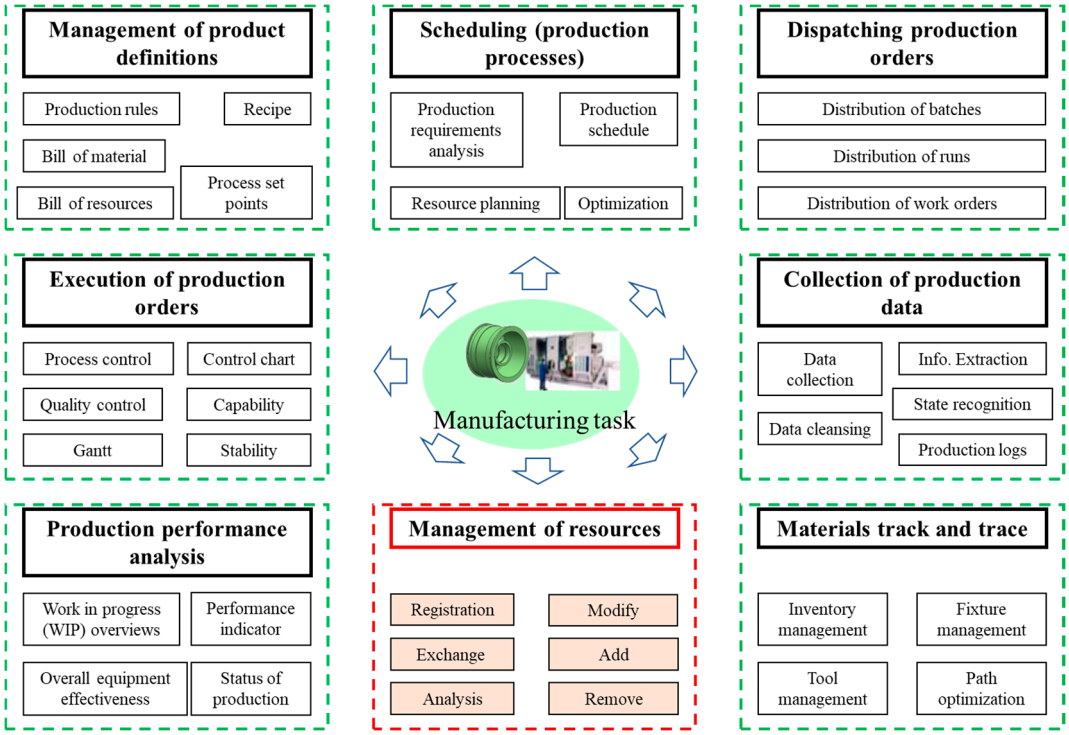



**Figure 6.** Functional decomposition of manufacturing task.

### 3.2.4. APIs for Data Pulls between M$^2$ES Apps

As mentioned above, M$^2$ES Apps can request and pull data from Basic DB (Graph DB) and other Apps' DBs (Relational DB) through the unified APIs. So, establishing reasonable APIs with the complete and reliable protocol is the basis for determining the efficient operation of the M$^2$ES Apps system. Since the order of development of the Apps is different, the new App must be compatible with the old ones when developing. Specific to the database and business logic design, developers of each App must obey the established naming rules when defining a DB. And the DB development document should be shared out for the later developers. Basic DB in the centralized server room stores some basic information in the form of graph database, and the data Table of each DB should be named as "*appName_tableName*". Noted that only the processing data can be saved in App DB.

## 4. Case Study

To demonstrate how to implement the M$^2$ES Apps System in actual industrial scenarios, this section reports a real-world case on designing, developing, deploying, and operating an M$^2$ES Apps system. The system in this case comes from an industry chain combined with several manufacturers, and it aims to achieve collaborative quality control in the level of a cross-workshop.

### *4.1. Industrial Background*

Figure 7 shows the main production process of Component T (seen in Figure 8) in an industry chain, and there are 9 related workshops in association with this simple component. *Assembly Plant* plays the role of the Customer for *Component Center*, which is the core of the entire industry chain.

*Component Center* has three independent departments, namely *Management*, *QC station*, and *Storeroom*. Besides, two suppliers *A* and *B* provide standard parts and raw materials, respectively, and three outsourcing factories *A*, *B*, and *C* provide manufacturing services for *Component Center*. These nine located in different places, so it is meaningful to design an MES to realize the information sharing and management of the whole production process.

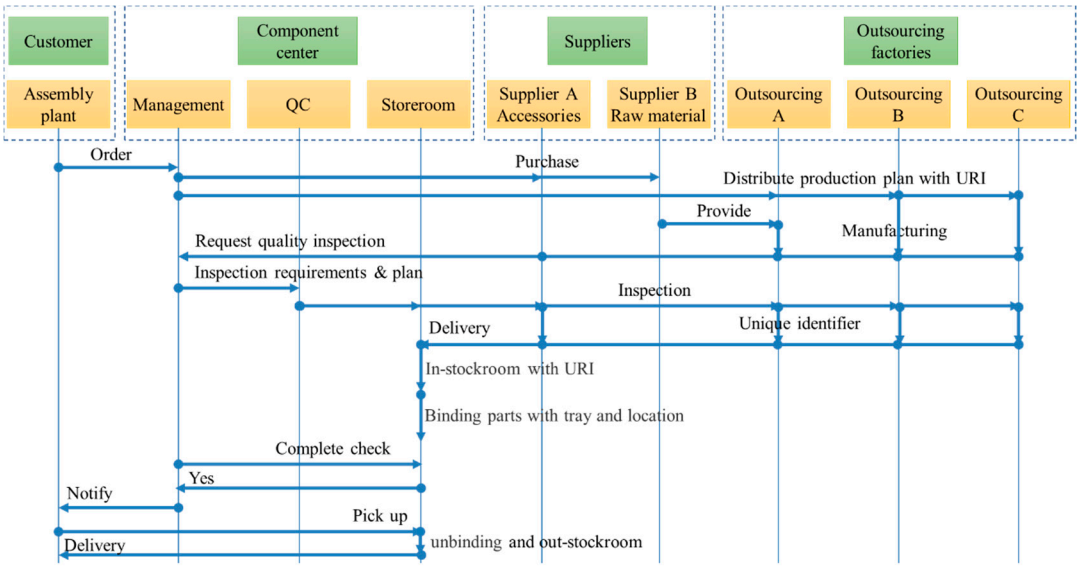



**Figure 7.** The production process of Component T in the industry chain.

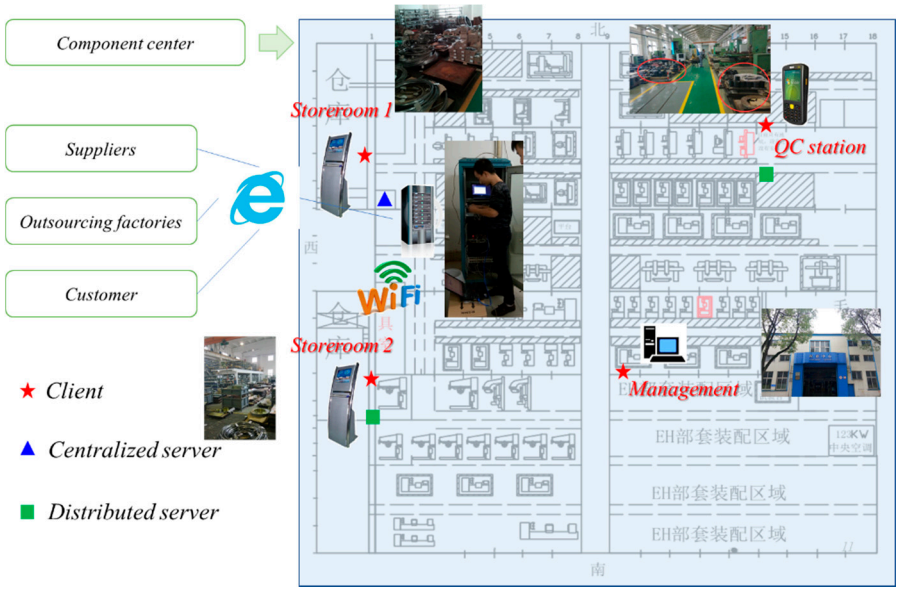



**Figure 8.** Hardware layout and photos.

Prior to the implementation of the M$^2$ES Apps System, the SAP system managed the relevant planning tasks. SAP was only accessible within the *Component Center*, production plans and controls relating to external suppliers and outsourcers are with paper or E-mail. At the same time, inventory management was based on an isolated self-developed program that only has basic functions of the inbound and outbound storage. And it runs in a single PC, which obviously cannot share data with SAP. Moreover, the inspection process was also based on the paper documents, and an NCR (Non-Conformity Report) system was operated independently.

### 4.2. Architecture Configuration

The architecture configuration can be mainly divided into Hardware layout optimization, Manufacturing resource description, and Software environment configuration.

#### 4.2.1. Hardware Layout Optimization

Firstly, the networking of all resources is realized, in which the WIFI-based industrial Internet was built in the *Component Center*, and the existing Internet and Internat can be used outside. The centralized server was placed in the computer room, and two distributed servers for deploying M$^2$ES Apps were added to it. Additionally, some clients were deployed where needed. The hardware layout and photos can be seen in Figure 8.

#### 4.2.2. Manufacturing Resource Description

Based on the generic ontology model, a domain ontology for static resources-cantered resources was created, as presented in Figure 9. Entities need to be tagged in the M$^2$ES Apps System with URI, so it is also important to define the encoding rules (Bar code).

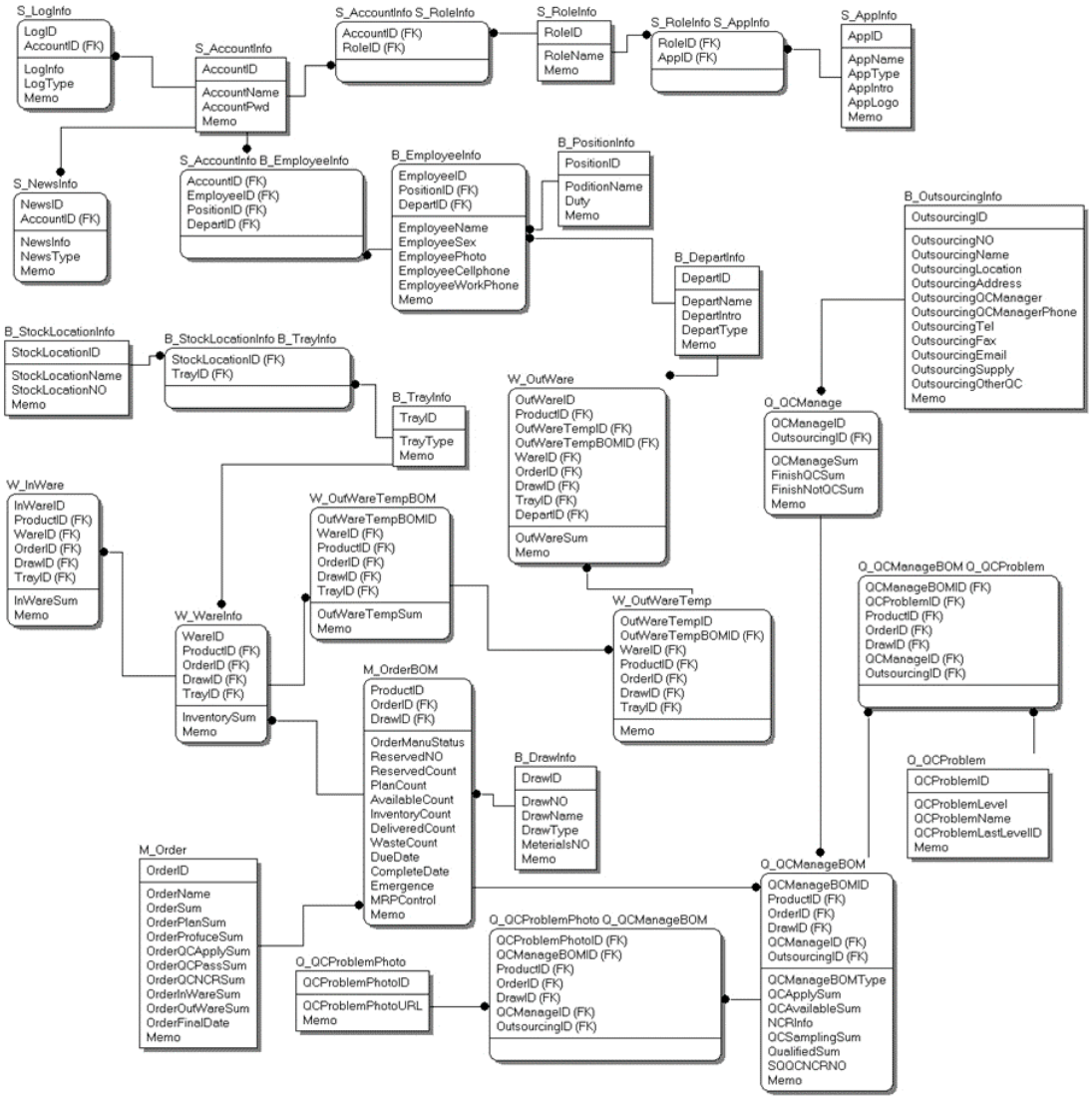



**Figure 9.** An ontology model for resources.

#### 4.2.3. Software Environment Configuration

As mentioned above, there are two major kinds of servers in the M$^2$ES Apps system. The one is the Centralized server for the platform, which is configured in the centralized server room. The other is the M$^2$ES App servers, which is configured in *Storeroom 2* and *QC station*. The centralized server mainly deploys services such as M$^2$ES App Store, M$^2$ES App Authorization, and Graph DB Management Systems (DBMS). App servers are supposed to deploy separate Apps, and only two servers were used in this case due to the small numbers of Apps. It should be noted that the B/S architecture was selected in order to enable the clients to use the M$^2$ES Apps easily.

### *4.3. Function Decomposition and M$^2$ES App Development*

According to Section 4.1, the M$^2$ES Apps System mainly implements the functions shown in Figure 10 in this case, and the corresponding M$^2$ES App can be designed and developed. There are 9 Apps totally in the first phase, each of them has relatively simple and strictly independent functions. And each App has its own isolated databases. Production and QC related Apps were deployed in the server in the *QC station*, and Inventory and Kanban related Apps were in *Storeroom 2*.

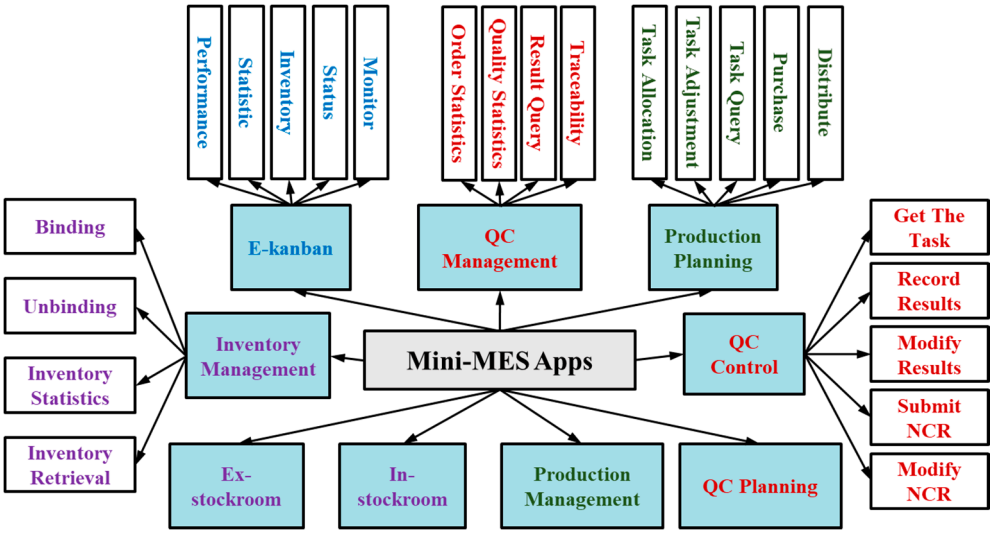



**Figure 10.** Function decomposition and corresponding M$^2$ES Apps.

Each App has a B/S architecture that follows a unified style on code and UI. Data can be requested among them through the API, and they can also access and modify the Graph DB located in the computer room through the DBMS.

### *4.4. Running of M$^2$ES Apps System*

What M$^2$ES Apps System presents to the user is an M$^2$ES App Store after configuring. Users can select Apps as needed to assist in solving production problems. Figure 11 demonstrates the user interface and data interaction of the In-stockroom App. It should be noted that this App can retrieve data from other Apps (QC App in the case) but cannot modify or save new-generated data in other Apps' DBs.

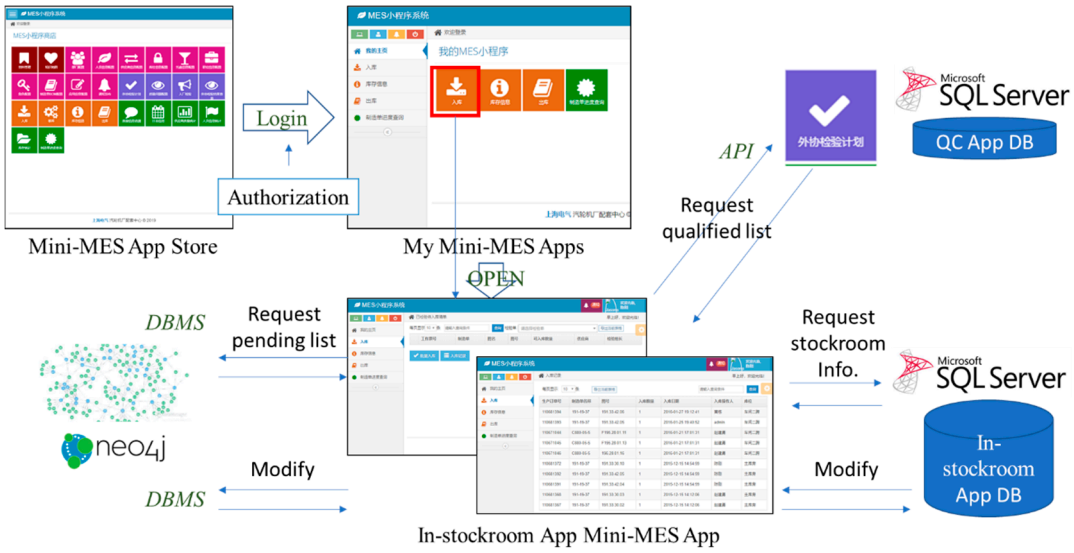



**Figure 11.** User interface and data interaction of the In-stockroom App.

### *4.5. Benefits Analysis of M²ES Apps System*

The M²ES Apps system has obvious architecture benefits compared to the traditional MES in dealing with the distributed manufacturing. This section will detail the performance, reliability, and scalability of the system on the basis of the proposed case.

Performance of the M²ES Apps System mainly depends on System deployment time and Response time. A comparative experiment has been conducted for measuring the above metrics when deploying Apps between centralized and distributed under the same network, and the results are shown in Figure 12. It can be seen that the distributed architecture is superior in both Total System Deployment Time and Total Response Time. It can also be seen that the deployment time is shorter due to the multiple servers, and the response time is shorter on account of the shorter network routes between the server and the client.

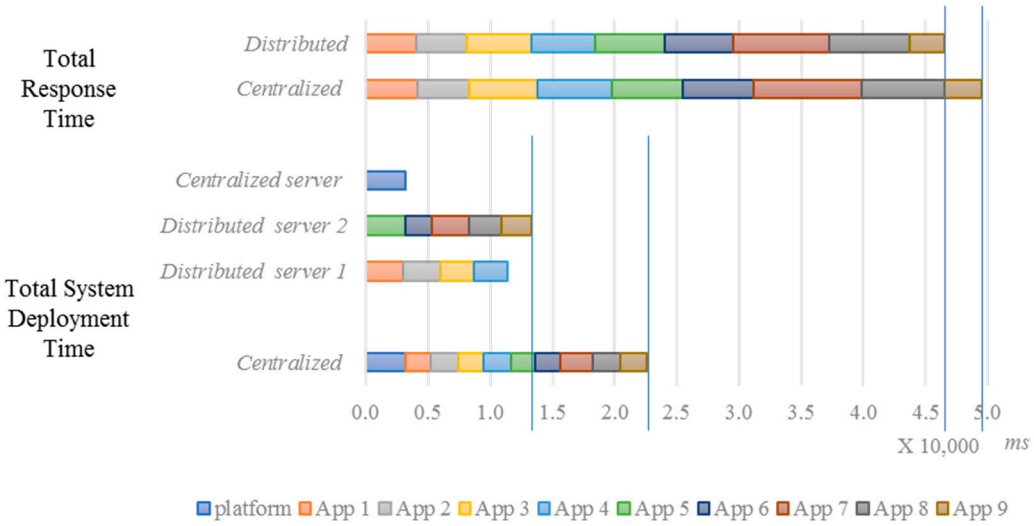



**Figure 12.** Experimental results on comparison of different architectures.

Reliability of the M²ES Apps system is mainly reflected in the fact that it will only affect limited Apps when a disturbance causes a node to out of service. Geographically separated and microservice-based architecture has enhanced the robustness of system greatly, and it has a greater tolerance for the crash or error reporting of a single App. Besides, the design of the independent

App is good for authority management, where it can be achieved by just assigning different Apps to different users.

Scalability of the M$^2$ES Apps system mainly focuses on the problem of frequent changes or adjustments in the current production. There are two types of changes normally. The first one is planned and controllable, such as the introduction of new products, the addition of new production lines or equipment, the changes in business processes and so on. Another change is unpredictable, such as the change of delivery time or quantity, the sudden interruption of production equipment, the quality problems of materials, and so on. The advantage of implementing an M$^2$ES Apps system is that no matter changes are needed, developers only need to pay attention to the new business logic and design, and develop some supplementary M$^2$ES Apps, without focusing on the compatibility issues. For the users, M$^2$ES Apps system can also adapt to more usage scenarios by using different arrangements and combinations of Apps.

## 5. Discussions

With the rapid changing of production, quite a few enterprises are facing the music on upgrading the existing information system in order to realize the flexible management in new production paradigm. The emergence of the M$^2$ES Apps system seems to address this thorny problem. The M$^2$ES Apps system has distributed and decentralized architecture, which can achieve extensible ongoing development. And its lightweight App logic enables it to have the ability to handle more complex dataspace in the level of cross-workshop. Furthermore, its microservices-based architecture also makes the M$^2$ES Apps system more efficient, reliable, and scalable.

In many industrial cases, the specialized or single-featured system offers a solution that is superior to an integrated one. But the question is how to make them interconnected easily because more integrations always go together with higher costs. M$^2$ES App, which is easy-to-develop, easy-to-deploy, easy-to-use, and easy-to-maintain, provides a sustainable information system architecture. Its authorization mechanism can also reduce the cost of use effectively under the premise of ensuring the security of data. In addition, decentralization of the M$^2$ES Apps drives localized saving of sensitive data, which also reduces the possibility of data leaking or tampering in the case of cross-domain usage.

In fact, thanks to blockchain technology, there might be more than one centralized server in M$^2$ES Apps system, without affecting the reliable synchronization of data. In other words, any companies in the industry chain can establish a self-centralized server as long as the data it owns is legal and trustworthy. The centralized servers can use the Distributed Ledger to maintain the consistent and untamed of data. In this way, the M$^2$ES Apps System will have a four-tier architecture, which includes the Distributed Ledger, the authorization and resources and data server, the M$^2$ES App server, and the user terminal.

## 6. Conclusions

A future-oriented MES called M$^2$ES Apps System is proposed here to play the role of a manufacturing ecosystem by being combined with ERP, CRM, and other information systems. M$^2$ES Apps System has a microservice-based feature and uses lightweight and flexible architecture to solve more complex logic functions in intelligent manufacturing. Users only need to Open, Use, Share, and Close its M$^2$ES Apps to assist production without uploading or downloading from the software.

This paper focuses on some basic properties and applications of M$^2$ES Apps System. M$^2$ES Apps System is developed as the software supporting more complex production situations. It is also an information system in connection with new manufacturing paradigms and production models. Then, a real industrial case using M$^2$ES Apps System is analyzed, and the result shows that it has better performance, reliability, and scalability than traditional MES.

The value of M$^2$ES Apps System lies in the deep integration of intelligent manufacturing and the industrial Internet, which can promote the transformation of systems to become digitalized, networked,

and intelligent. In this process, M$^2$ES Apps System not only promotes enterprises to build new platforms, but also promotes the generation of new economies, new paradigms, and new ecosystems.

**Author Contributions:** Conceptualization, P.L. and P.J.; Formal analysis, P.L.; Methodology, P.L.; Software, P.L. and J.L.; Supervision, P.J.; Validation, J.L.; Writing—original draft, P.L.; Writing—review & editing, P.L.

**Funding:** The research work is under the financial supports of both MOST's innovation method working projects with Grant No. 2016IM010100 and natural science foundation of China with Grant No. 71571142.

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

## References

1. Qiu, R.G.; Zhou, M. Mighty MESs; state-of-the-art and future manufacturing execution systems. *IEEE Robot. Autom. Mag.* **2004**, *11*, 19–25. [CrossRef]
2. Tao, F.; Qi, Q.; Liu, A.; Kusiak, A. Data-driven smart manufacturing. *J. Manuf. Syst.* **2018**, *48*, 157–169. [CrossRef]
3. SAP Manufacturing Execution System Software. Available online: https://www.sap.com/uk/products/execution-mes.html (accessed on 8 August 2019).
4. Oracle Drive Manufacturing Excellence. Available online: https://www.oracle.com/uk/applications/supply-chain-management/solutions/manufacturing/discrete-manufacturing.html (accessed on 8 August 2019).
5. Siemens Manufacturing Execution System (MES)—Respond in Real Time. Available online: https://w3.siemens.co.uk/automation/uk/en/manufacturing-execution-system-mes/pages/default.aspx# (accessed on 8 August 2019).
6. Helo, P.; Suorsa, M.; Hao, Y.; Anussornnitisarn, P. Toward a cloud-based manufacturing execution system for distributed Manufacturing. *Comput. Ind.* **2014**, *65*, 646–656. [CrossRef]
7. Zhou, J.; Li, P.; Zhou, Y.; Wang, B.; Zang, J.; Meng, L. Toward New-Generation Intelligent Manufacturing. *Eng. PRC* **2018**, *4*, 11–20. [CrossRef]
8. Ye, F.; Wang, Z. Effects of information technology alignment and information sharing on supply chain operational performance. *Comput. Ind. Eng.* **2013**, *65*, 370–377. [CrossRef]
9. Stone, A.; Dennis, J.; Strout, M. Establishing a Miniapp as a programmability proxy. *SIGPLAN Not.* **2012**, *47*, 333–334. [CrossRef]
10. Nandi, M.L.; Kumar, A. Centralization and the success of ERP implementation. *J. Enterp. Inf. Manag.* **2016**, *29*, 728–750. [CrossRef]
11. Jiang, P.; Li, P. Shared factory: A new production node for social manufacturing in the context of sharing economy. *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.* **2019**. [CrossRef]
12. Li, J.; Tao, F.; Cheng, Y.; Zhao, L. Big Data in product lifecycle management. *Int. J. Adv. Manuf. Technol.* **2015**, *81*, 667–684. [CrossRef]
13. Huang, M.; Rust, R.T. IT-Related Service: A Multidisciplinary Perspective. *J. Serv. Res.* **2013**, *16*, 251–258. [CrossRef]
14. Jiang, P.; Zhang, C.; Leng, J.; Zhang, J. Implementing a WebAPP-based Software Framework for Manufacturing Execution Systems. *IFAC PapersOnLine* **2015**, *48*, 388–393. [CrossRef]
15. Tencent WeChat Mini Programs. Available online: https://mp.weixin.qq.com/cgi-bin/wx?tlang=en_US (accessed on 22 May 2019).
16. Hao, L.; Wan, F.; Ma, N.; Wang, Y. Analysis of the Development of WeChat Mini Program. In Proceedings of the First International Conference on Advanced Algorithms and Control Engineering (ICAACE 2018), Taiwan, China, 10–12 August 2018; Volume 1087, p. 62040.
17. Ding, K.; Jiang, P. Social Sensors (S2ensors): A Kind of Hardware-Software-Integrated Mediators for Social Manufacturing Systems Under Mass Individualization. *Chin. J. Mech. Eng.* **2017**, *5*, 1150–1161. [CrossRef]
18. Ding, K.; Jiang, P. Incorporating social sensors, cyber-physical system nodes, and smart products for personalized production in a social manufacturing environment. *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.* **2018**, *13*, 2323–2338. [CrossRef]
19. Wikipedia Multi-Agent System. Available online: https://en.wikipedia.org/wiki/Multi-agent_system (accessed on 20 September 2018).

20. Zhong, R.Y.; Xu, X.; Klotz, E.; Newman, S.T. Intelligent Manufacturing in the Context of Industry 4.0: A Review. *Eng. PRC* **2017**, *3*, 616–630. [CrossRef]
21. Jiang, P.; Ding, K.; Leng, J. Towards a cyber-physical-social-connected and service-oriented manufacturing paradigm: Social Manufacturing. *Manuf. Lett.* **2016**, *7*, 15–21. [CrossRef]
22. Almada-Lobo, F. The Industry 4.0 revolution and the future of manufacturing execution systems (MES). *J. Innov. Manag.* **2016**, *3*, 16–21. [CrossRef]
23. Wang, S.; Wan, J.; Zhang, D.; Li, D.; Zhang, C. Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. *Comput. Netw.* **2016**, *101*, 158–168. [CrossRef]
24. Zhang, Y.; Huang, G.Q.; Sun, S.; Yang, T. Multi-agent based real-time production scheduling method for radio frequency identification enabled ubiquitous shopfloor environment. *Comput. Ind. Eng.* **2014**, *76*, 89–97. [CrossRef]
25. Malaurent, J.; Yan, L.; Avison, D.E. Reopening the Centralization-Decentralization Debate: A Comparative Case Study of ERP Implementation in Two Chinese Petroleum Companies. In Proceedings of the Pacific Asia Conference on Information Systems (PACIS) 2012, Ho Chi Minh City, Vietnam, 11–15 July 2012; p. 74.
26. Asmae, A.; Souhail, S.; El Moukhtar, S.; Hussein, B. Using ontologies for the integration of information systems dedicated to product (CFAO, PLM . . . ) and those of systems monitoring (ERP, MES..). In Proceedings of the 2017 International Colloquium on Logistics and Supply Chain Management (LOGISTIQUA), Ensias, Rabat, Morocco, 27–28 April 2017; pp. 59–64.
27. Strozzi, F.; Colicchia, C.; Creazza, A.; Noè, C. Literature review on the 'Smart Factory' concept using bibliometric tools. *Int. J. Prod. Res.* **2017**, *55*, 6572–6591. [CrossRef]
28. Theorin, A.; Bengtsson, K.; Provost, J.; Lieder, M.; Johnsson, C.; Lundholm, T.; Lennartson, B. An event-driven manufacturing information system architecture for Industry 4.0. *Int. J. Prod. Res.* **2017**, *55*, 1297–1311. [CrossRef]
29. Salah, T.; Zemerly, M.J.; Chan, Y.Y.; Al-Qutayri, M.; Al-Hammadi, Y. The evolution of distributed systems towards microservices architecture. In Proceedings of the 11th International Conference on Internet Technology and Secured Transactions, Barcelona, Spain, 5–7 December 2016; pp. 318–325.
30. Wang, X.; Sun, X.; Hao, Z.; Li, H. Research on the Construction of Resource Sharing Platform Based on MicroService. In Proceedings of the 2017 International Conference on Smart Grid and Electrical Automation (ICSGEA), Changsha, China, 27–28 May 2017; pp. 713–717.
31. Cao, R.; Lu, S.; Wang, X.; Xiao, H.; Chi, X. Unified Account Management for High Performance Computing as a Service with Microservice Architecture. In Proceedings of the International Symposium on Grids and Clouds (ISGC) 2018, Taipei, Taiwan, 16–23 March 2018.
32. Ciavotta, M.; Alge, M.; Menato, S.; Rovere, D.; Pedrazzoli, P. A Microservice-based Middleware for the Digital Factory. *Procedia Manuf.* **2017**, *11*, 931–938. [CrossRef]
33. Kleanthi, T.; Vachtsevanou, D.C.; Solano, A. Cyber-physical microservices: An IoT-based framework for manufacturing systems. In Proceedings of the 2018 IEEE Industrial Cyber-Physical Systems (ICPS), St. Petersburg, Russia, 15–18 May 2018; pp. 232–239.
34. Franklin, M.; Halevy, A.; Maier, D. From databases to dataspaces: A new abstraction for information management. *SIGMOD Rec.* **2005**, *34*, 27–33. [CrossRef]
35. Mirza, H.T.; Chen, L.; Chen, G. Practicability of dataspace systems. *Int. J. Digit. Content Technol. Appl.* **2010**, *4*, 233–243.
36. Monostori, L.; Kádár, B.; Bauernhansl, T.; Kondoh, S.; Kumara, S.; Reinhart, G.; Sauer, O.; Schuh, G.; Sihn, W.; Ueda, K. Cyber-physical systems in manufacturing. *CIRP Ann.* **2016**, *65*, 621–641. [CrossRef]
37. Angrish, A.; Starly, B.; Lee, Y.; Cohen, P.H. A flexible data schema and system architecture for the virtualization of manufacturing machines (VMM). *J. Manuf. Syst.* **2017**, *45*, 236–247. [CrossRef]
38. Xiao, G.; Calvanese, D.; Kontchakov, R.; Lembo, D.; Poggi, A.; Rosati, R.; Zakharyaschev, M. Ontology-based data access: A survey. In Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI-18), Stockholm, Sweden, 13–19 July 2018; AAAI Press: Menlo Park, CA, USA, 2018; pp. 5511–5519.
39. McHugh, J.; Cuddihy, P.E.; Williams, J.W.; Aggour, K.S.; Kumar, V.S.; Mulwad, V. Integrated access to big data polystores through a knowledge-driven framework. In Proceedings of the IEEE 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 11–14 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1494–1503.

40. Kharlamov, E.; Mailis, T.; Mehdi, G.; Neuenstadt, C.; Özçep, Ö.; Roshchin, M.; Solomakhina, N.; Soylu, A.; Svingos, C.; Brandt, S.; et al. Semantic access to streaming and static data at Siemens. *J. Web Semant.* **2017**, *44*, 54–74. [CrossRef]
41. Oh, S.; Kim, Y.; Cho, S. An Interoperable Access Control Framework for Diverse IoT Platforms Based on OAuth and Role. *Sensors (Basel)* **2019**, *19*, 1884. [CrossRef] [PubMed]
42. Golodoniuc, P.; Car, N.; Cox, S.; Atkinson, R. PID Service—An advanced persistent identifier management service for the Semantic Web. In Proceedings of the 21st International Congress on Modelling and Simulation (MODSIM2015), Broadbeach, Queensland, Australia, 29 November–4 December 2015.
43. Lu, Y.; Xu, X. Resource virtualization: A core technology for developing cyber-physical production systems. *J. Manuf. Syst.* **2018**, *47*, 128–140. [CrossRef]