Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 81 (2019) 892–897

www.elsevier.com/locate/procedia

52nd CIRP Conference on Manufacturing Systems

A hybrid framework for industrial data storage and exploitation

K. Grevenitis[a], F. Psarommatis[b*], A. Reina[c], W. Xu[c], I. Tourkogiorgis[b], J. Milenkovic[a], J. Cassina[c], D. Kiritsis[b]

[a]*ATLANTIS Engineering S.A., 21 Ant. Tritsi str., Thessaloniki 57001, Greece*
[b]*École polytechnique fédérale de Lausanne, ICT for Sustainable Manufacturing, EPFL SCI-STI-DK, Lausanne, Switzerland*
[c]*Holonix srl, 8 Corso Italia str., Meda 20821, Italy*

* Foivos Psarommatis. Tel.: +41 21 69 31086; *E-mail address:* foivos.psarommatis@epfl.ch.

Abstract

In this paper a hybrid framework is illustrated, with a software and hardware integration strategy, for an industrial platform that exploits features from a Relational Database (RDB) and Triplestore using the blackboard architectural pattern, ensuring efficient and accurate communication concerning data transfer among software applications and devices. Specifically, "Raw Data Handler", manages unstructured data from IoT devices that are kept in an Apache Cassandra instance, while "Production Data Handler" acts on structured data, persisted in a MySQL database. Filtered data is transformed into knowledge and persisted into the Triplestore database (DB) and can be retrieved by expert systems at any time. The proposed framework will be tested and validated within Z-Fact0r project.

Keywords: industry 4.0; IoT; integration strategy; data exploitation and storage; Triplestore; production and sensorial data; zero-defect manufacturing

---

1. Introduction

The latest industrial revolution (Industry 4.0) has stepped forward quite a lot since industries upraise in 18th century that used water and steam power to mechanise production [1]. Today's industry, is part of Information and Communication Technology (ICT) industry, and applies the Cyber Physical Systems (CPS) philosophy. Sensors are used for data collection, while transcending to the Artificial Intelligence (AI) from an input and output approach to a smooth communication between humans and robots, evolution the whole production process, where a minor human involvement is required (when and if it is required) [2]. The game changer for this digital manufacturing as well as for intelligent production and process design [3] is the Internet of Things (IoT) which is associated with the Industry 4.0 concept and architecture.

Notwithstanding, the great developments brought by technological research have still various open issues for an effective and efficient implementation of Industry 4.0, following a philosophy that includes machines and devices that are continuously interconnected and communicating with each other and also with humans. One of those issues is the lack of interoperability and the difficulty to interface a variety of data sources in a fragmented world based upon proprietary and heterogeneous transmission protocols. An ontological framework can represent a good answer to address interoperability issues, but its huge potential shows some limitations when real time data is involved. Furthermore, all the optimisation processes in the contemporary manufacturing environment require stored data from multiple and different sources, thus this is a challenge [4] that requires more research to address these multiple issues [5].

In this context, the main objective of this paper is to present an innovative approach consisting of a hybrid framework for an industrial distributed software platform for Zero-Defect Manufacturing (ZDM), able to integrate the advantages of

2212-8271 © 2019 The Authors. Published by Elsevier Ltd.

10.1016/j.procir.2019.03.221

semantic modelling with real-time data managing. Thus, the best features of both relational databases and a Triplestore for triplets' storage are exploited, while structured and unstructured data handlers will ensure an efficient and accurate data transfer communication among the various software applications and devices. In order to enable a fast deployment of the proposed framework to support a rapid access to the market, a novel integration strategy has been implemented and is presented here.

2. State-of-the-art

According to Rajkumar [5], the CPS, as a set of "physical and engineered systems whose operations are monitored, controlled, coordinated and integrated by a computing and communicating core", are applied to smart manufacturing, especially within the Industry 4.0 concept, leaning on the heavy usage of IoT devices and creating a smart factory. A major problem in manufacturing plants in the past was the requirement of comprehensive and thorough knowledge of existing systems, incorporation of consolidated products and development or parameterisation of various kinds of adapters and gateways. This challenge led the integration of various devices and software applications at the shop floors to be difficult, costly and often impossible to achieve [6]. Recently, the general structure of an IoT system had been defined based on various layers [7] (e.g. communication, data, business logic, presentation layer) that require the proper integration process in order to avoid this problem. It is well known that in the past connectors were heterogeneous and their various interfaces were difficult to communicate, bringing additional burden to the system's integration implementation. Furthermore, in the past 30 years, a lot of different technologies and standards have been developed to ease the integration process itself (e.g. CORBA, COM), Service Oriented Architecture (SOA), RESTful, etc. Also, nowadays sensors and IoT devices support protocols such as HTTP (Hypertext Transfer Protocol) and IIOP (Internet InterORB Protocol) while data integration processes orchestrate the data flow and federation among heterogeneous data sources. These are deployed on different machines, under different operating systems and database management systems while dividing the integration into sub-layers (i.e. basic coordination, functional interfaces, business protocol/policies and non-functional properties) where the presentation layer integration focuses on the User Interface (UI) integration.

In general, the integration process, the IoT and Industrial IoT (IIoT) in particular, requires a variety of data, collected from various sources, setting the middleware layer of crucial importance by acting as a linking element between the physical and the cyber world, providing a resource pool with all kinds of operations, by hiding the complexity and the heterogeneity of underlying infrastructures [8]. An IoT middleware needs to provide data management services to applications, including data acquisition, data processing (including pre-processing) and data storage [9]. Research on IoT middleware architectures is still in its early stage; nonetheless, some IoT-specific typologies are emerging. Many solutions have been proposed and implemented, especially in the last couple of years that are highly diverse in their design approaches (e.g., event-based, database), level of programming abstractions (e.g., local or node level, global or network level), and implementation domains (e.g., WSNs, RFID, M2M, and SCADA). Extensive surveys are presented in [9-16] showing the possibility to categorise existing middleware solutions according to their design approach in event-based, service-oriented, agent-based, tuple-space, Virtual Machine (VM)-based, database-oriented and application-specific. The Service-Oriented and VM-based approaches support abstraction, network and application level scalability. The agent-based design approaches are good at resource and code management due to their mobile and distributed nature, as is the tuple-spaces, which is also more reliable because of their data redundancy characteristics. Both approaches weaknesses are related with data security and privacy.

On one hand, database design approaches perform well in data management, with quick response times, but they are not able to handle real-time responses. On the other hand, event-based middleware platforms are suitable for mobile and reactive applications, but have limited interoperability. Furthermore, knowledge management corresponds to a wide set of methods and techniques which are utilised in order to represent, distribute and reuse information, know-how and other [17] pointing out one more need of manufacturing environment today. In addition, it is noticed that a lot of companies do not know the exact knowledge that they pose within their databases [18]. For that reason and for increasing the level of interoperability among different systems that are using different data structures and various technologies, Semantic Web technologies, and in particular the Web Ontology Language (OWL) [19], have been developed and used for various applications in the manufacturing sector as presented in [20]. Furthermore, web technologies have been utilised in the context of factories and manufacturing systems that were addressed in European projects Linked Design [21] and Virtual Factory Framework [22]. Also, semantic repositories and inference engines have been improved dramatically over the last years allowing the efficient storage of data and reasoning in semantic data [23], extending the development of knowledge management systems. Another ontology advantage is the capability for integrating various data from different sources in one unique model [24]. The use of semantic technologies together with ontologies and inference engines helps creating detailed and accurate knowledge models providing the "side effect": the ability to identify new knowledge [20].

Service oriented architecture and possibly micro services and containerisation in the future represents, at the moment, the best approach for interoperability. The industrial sector is moving towards the utilisation of tools for knowledge management such as ontologies. In this context, solutions [25] are proposed, able to provide syntactic and semantic level interoperability using semantic web services. The main drawback is that real-time data cannot be handled with a semantic approach The concept proposed in [26] is to combine multiple data sources [8], using a flexible data storages to provide the ability to properly manage multiple collected data that are heterogenous, with different structures while the storage component categorises and organises data in a transparent way [27]. This flexible approach is considered as a starting point for the novel hybrid approach proposed hereafter, whose aim is to overcome the limitation of semantic framework to handle real-time data, guaranteeing at the same time increased interoperability and providing the capability to identify new knowledge. Additionally, current integration technologies are the Remote Method Invocation (RMI), messaging and message brokers, queuing systems, transaction management systems, adapters and database connectivity solutions [6]. Furthermore, a number of Enterprise Integration methodologies have been introduced (i.e. the Purdue Enterprise Reference Architecture (PERA), the Open System Architecture for Computer Integrated Manufacturing (CIMOSA), GERAM [28], QUASAR [29]) that are used up to today. It has to be pointed out that an integration approach is important to handle the business aspects of a company and their enterprise strategies (e.g. mission, vision, values, critical success factors, business strategies) as well as the effect on human resources.With respect to the currently implemented approaches, one of main advantages and innovations of the architecture model proposed hereafter, are its flexibility and the possibility to be implemented starting from state of the art communication and computing technologies, but keeping the focus on cloud services.

3. Proposed Framework

Within this work, an innovative architectural framework is proposed with the goal to improve the data retrieval and management process in order to have an efficient, user friendly and easily deployable platform to be widely used by industrial companies for ZDM. This is achieved by integrating two different technologies for data storage while exploiting the best features from each of them. Its advantages in terms of easy implementation and adoption are also demonstrated in industrial environments to prove the feasibility of the concept and to set a roadmap towards scalability.

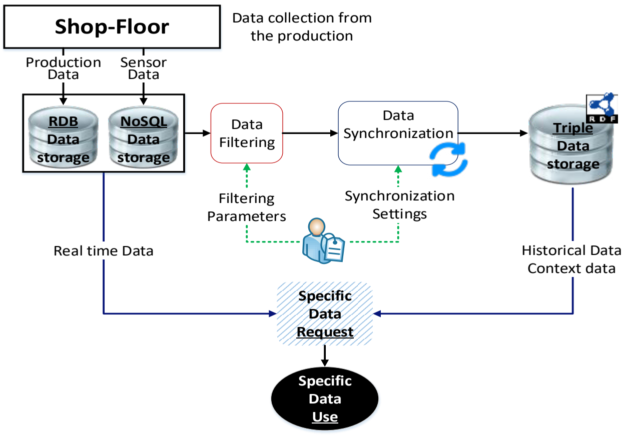

Fig. 1. Proposed approach Hybrid Framework.

The main concept is that in a generic manufacturing company, there are various data streams from the factory shop floor, from sensors or machines, from legacy system or through other IoT devices. The data flow is transmitted via an Enterprise Service Bus (ESB) that implements a distributed system through a communication sequence among mutually interacting software applications in a Service-Oriented Architecture (SOA), where any application can behave as server or client in turns. ESB promotes agility and flexibility and its fundamental goal is the Enterprise Application Integration (EAI) of heterogeneous and complex systems.

In the proposed framework, structured data are stored in relational database systems, such as MySQL. Sensorial data that tend to be unstructured or semi structured data is stored into NoSQL column-oriented DB systems such as Apache Cassandra, which is particularly suitable for distributed environments and big size cluster. User defined filters are applied on the data, transforming it into Resource Description Framework (RDF) triples that are synchronised in real time with a Triplestore making the data available to the user.

Apache Jena is the framework of choice and TDB (Triplestore database), its component for RDF storage and query, the Triplestore. This choice has been guided by the fact that it is an open source framework for building Semantic Web and Linked Data applications and widely used by the scientific community. Consequently, the presented framework at Fig.1 aims to achieve a high-level flexibility by satisfying the requirements of heterogeneous production environments, considering the integration of heterogeneous data produced by various sources and different levels consumed by software with different needs. It also supports full text search by combining the technologies of Apache Lucene and Elastic Search.

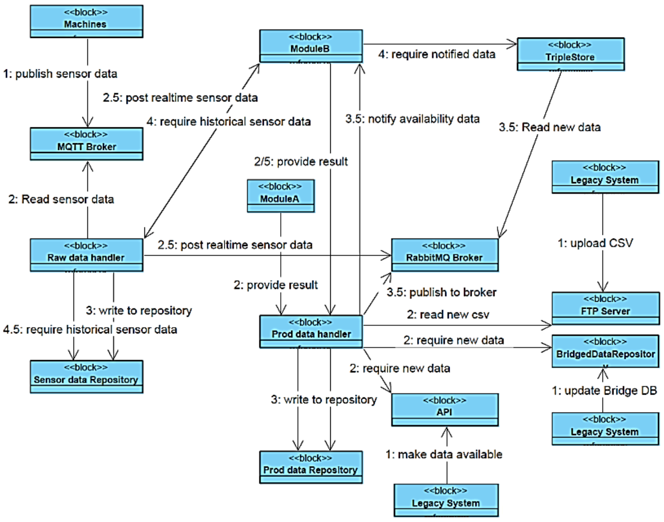

Fig. 2. Component diagram of the proposed solution.

3.1. Middleware & Communication

In a production plant, data is generated by various sensors installed on production systems and/or nearby, integrated or not in a Programmable Logic Controller (PLC), from company legacy systems, such as Manufacturing Execution System/Enterprise Resource Planning (MES/ERP) and/or from various expert systems that also act as data consumers. In order to manage the complexity of the interactions among data producers and consumers, a proposed data flow is presented in Fig. 2, where the key elements of this framework are the Raw Data Handler that manages sensors data and the Production Data Handler that deals with structured data.

The real time data from sensors is published to a MQTT (Message Queueing Telemetry Transport) broker that is a machine-to-machine connectivity, extremely lightweight publish/subscribe messaging transport, ISO standardised protocol that works on top of the TCP/IP protocol. It is suitable to be used to connect with remote locations, where a small code footprint is required and/or bandwidth is limited [31].

The Raw Data Handler subscribes to that MQTT broker and acquires sensorial data that later is persisted to the sensorial data repository where the data is indexed and thus it is easily filterable by timestamp values. The next step is to send that data to a real time data consumer and finally pushing it to the Triplestore through the message queue. Any other software that needs access to the historical data can retrieve them from the Triplestore, while the legacy systems data can be provided through APIs, a DB bridge or a File Transfer Protocol (FTP) server.

The Production Data Handler performs a scheduled data request that gathers the data from the relational database, and pushes it further through the message queue to the Triplestore DB. Furthermore, the last category of data sources is represented by applications that can produce data as files or not. A produced file is sent to the Production Data Repository through the Production Data Handler where a unique identification (ID) is assigned to the file and the meta information is sent to Triplestore DB. Subscribed software applications are notified that a new file is available and can be retrieved with its unique ID.

An application can also produce unstructured data that flows to the Production Data Handler, and then to the Production Data Repository. RESTful APIs over HTTP have been chosen for that communication since its architectural style provides interoperability among online available systems, while the HTTP protocol is the foundation of data communication for the World Wide Web [32]. Also, it has been chosen due to the higher level of complexity of information exchange.

In the Production Data Repository, the unstructured data is persisted in relational. The relational data is pushed through the message queue to the Triplestore DB where it can be retrieved by any other applications and notify any subscribed applications that new data has been pushed to the Triplestore database.

3.2. Triplestore and Synchronisation mechanism

Among the different DB, the synchronisation mechanism is of primary importance.

The expected system behaviour is a real-time synchronisation mechanism between a relational database and an RDF Triplestore. The system is notified of any data changes in the relational database and acts accordingly resulting in an up-to-date synced RDF Triplestore.

The Synchronizer Service (Data Repository Synchronizer Service) is the service that process incoming RabbitMQ messages and update the Triplestore, the database for storing the relational data as RDF triples.

When data in the relational database is changed, RabbitMQ will publish a relevant message containing all the necessary information. Data Synchronizer Service is a process that runs continuously and consumes these specific messages, by subscribing to the relevant RabbitMQ channel. The Data Synchronizer Service logic is implemented following the data flow programming where the Data Synchronizer Flow consists of three flow steps. The Receive Step, responsible for connecting to RabbitMQ, receives published messages and push them to the next step, RDFizer Step. Furthermore, this will transform the received message to RDF triples (second step) based on a provided Ontology, and push the transformed message to the final, Persist Step, where the messages as RDF triples will be saved in the Triplestore. Users or any client system are able to query the data through an exposed SPARQL endpoint. In addition, based on the specific Triplestore technology, more semantic tools could be applied on data for further exploitation (inference, etc.).

3.3. Integration strategy of software applications and devices

In order to effectively implement the proposed framework an Incremental Integration Strategy (IIS) is proposed. It provides a unified framework and a methodology with a number of monitored factors that follow the philosophy of “early and frequent” approach. This approach manifests that the software applications are integrated and tested incrementally by assembling sub systems till the total solution is integrated in the end. It is influenced by the bottom-up integration approach where “lower-level” software applications (that are closer to the production line) are integrated first and later the “higher-level” (business logic) software applications . This approach enables uncovering quite early any problems that might occur and then to be solved through each iteration. The strategy prioritises the tests execution by a number of factors and a time schedule that is also defined.

The strategy’s goal is to be completed in a timely manner with respect to the business needs and human resources. The responsibilities are assigned to a number of teams that work sequentially. Each team runs the iterations of the addressed integration tests. After each iteration is completed, a number of results reports are generated. Then the results are evaluated by quality assurance (QA) team, and if the iteration fails, the integration scenarios are updated, possible bugs are fixed, and the systems are deployed again for testing. Following the strategy above, the test conditions are easier to create while the test results monitoring is better.

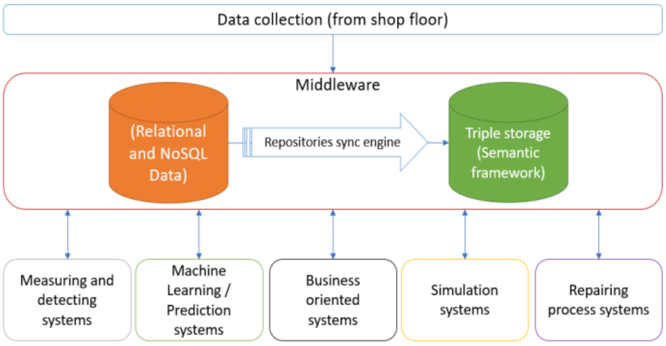

Fig. 3. Proposed framework application to Industrial use cases.

4. Industrial application

In order to demonstrate the feasibility of the proposed framework shown in Fig. 1, two different industrial scenarios were used, where the main objective is to develop a zero-defect manufacturing solution. The use cases fall under the manufacturing domains of hard metal parts and semi-conductors. Respectively, the first pilot scenario is analysing the multi-stages process to produce a hard metal parts from powder preparation through green machining, sintering and grinding, while the second pilot scenario, case considers the single stage process of glue deposition in a Printed Circuit Board (PCB) assembly. Furthermore, a variety of different legacy software applications have been used in both cases, but not all of them are communicating to each other (in a direct or indirect way) for data exchange (or if they do, they do it in a semi-automatic way). In order to avoid a direct access to those heterogeneous systems, as the proposed framework described in section 3, a common data message format is agreed upon and RESTful APIs are implemented and used as communication proxies. The flexibility requirements of such an approach make the above scenarios suitable to validate the proposed framework. Additionally, sensors have been installed to monitor critical variables in specific pilot scenarios such as: vibrations in case of green machining in the production of hard metal parts, nozzle pressure and glue temperature in the use case of the PCB manufacturing process.

In the validation scenarios the Modules depicted in Fig.2 are software components that enable the application of the ZDM strategy, in terms of detection, prediction, prevention and repair, as explained below.

The software components and the devices were integrated by following the Incremental Integration Strategy (IIS) presented in sub-section 3.3. After the integration process is successfully completed, the final distributed system is up and running from end to end in the business scenarios (Fig. 3). The system collects data from various shop floor data sources. The data is divided into two categories, production (historical) data and sensorial (real time) data. On one hand, production data sources are legacy systems, while on the other hand, the sensorial data is collected from PLCs and IoT devices. After the collection, data is published to a message queue and persisted to a NoSQL database to be later consumed by the triple data storage application in order to be filtered, synced and persisted as RDF data. The main objective is to exploit all the benefits from this semantic approach. The final step is the RDF data to be available to the rest software applications (expert systems).

The middleware supports two data repositories: one NoSQL repository and a relational one. The Semantic Framework is an integrated software stack using semantic technologies for knowledge management. It has a layered architecture that combines existing open source software with additional open source components developed specifically to provide a complete semantic technology framework. The premise of the entire stack is based on the RDF data model.

The measuring and detecting systems with the simulation systems need real time data, repair processes systems require historical data, while Machine Learning/prediction systems and business-oriented Systems need data of both data streams. Each expert system is a knowledge source, responsible for storing current states and producing new knowledge solving specific problems.

4.1. Industrial use cases with the proposed framework application

In detail, to show the feasibility of the proposed framework the ZDM platform, based on the architecture presented in Fig. 3, has been deployed in semi-conductor industry to evaluate defected PCB that can be affected with variable volumes of glue (less or excess) during the production. The glue volume data measured by the detecting application, is sent to the middleware and stored in the Triplestore DB. At the same time machine and sensors data are acquired and managed by the Raw Data Handler. The context aware software runs prediction algorithms that exploit and classify historical data from the Triplestore DB and correlate them with sensorial ones from the machines (pressure, temperature, etc.) with the goal to be able, in future stages, to prevent the defect to happen. The defected PCB is temporarily excluded from the production line. If the repair/rework expert system (part of business-oriented systems) in real time justifies any repairing cost, while there is enough value to recover the circuit, the PCB is repaired and sent back to the production line again. The produced knowledge is sent back to the middleware where the Production Data Handler takes care of persisting it in the Triplestore, in order to be used by the other expert systems.The end to end scenario has been validated by demonstrating that the approach proposed is feasible, realistic and allow an easy and fast deployment of the platform in industrial environment.

In the framework of hard metal industry the ZDM platform based on the architecture illustrated on this paper is applied to evaluate components realized with green machining. After the production process is completed, an automatic quality check is performed by the detection module. The quality control results are sent to the middleware, managed by the production data handler and stored in MySQL repository and in the Triplestore DB. At the same time, the sensorial data from machines are handled by the Raw Data Handler, stored in the NoSQL DB and copied into the Triplestore DB to be transformed into historical data. Results are then used by Machine Learning software applications that generate short term predictions for near time defects. Similarly, to the PCB use case, if any repairing cost is justified, and if there is enough value to repair the defected part, it is repaired by the repairing process systems and the produced knowledge is sent to the middleware.

Also, in this case, the end to end scenario has been validated by demonstrating that the approach proposed is feasible, realistic and allow an easy and fast deployment of the platform in industrial environment.

5. Conclusion

This paper emphasizes the need for a consistent framework for integrating various legacy heterogeneous information systems of a semi-automated production line, ensuring proper information flow and data transformation and bringing it closer to Industry 4.0.

An innovative hybrid architecture is thus proposed to combine the advantages in terms of interoperability provided by the adoption of a semantic framework with the need for real-time data consumption. Data storage is carried out accordingly, considering both relational and Triplestore DBs, integrated by following the Incremental Integration Strategy. Key elements in the communication flow are the Raw Data Handler to manage sensor related data and Production Data Handler.

The proposed framework has been applied in industrial environments to demonstrate its feasibility and its advantages in deploying a modular platform for ZDM.

Regarding the shop floors data collection, a further research will focus on the extension of 5C (Connection, Conversion, Cyber, Cognition, Configuration) Industrial Network reference architecture. 5C architecture can be applied to different levels of an industrial concern, including sensors, IoT devices, machines, fleets and the enterprise. Each level uses different analytics to generate useful information from raw data and useful knowledge about the system. The upper levels of the hierarchy use analytical methods to aggregate data from the lower levels [30].

Future work will focus on a deeper evaluation of system's performance and examine the scalability of the presented approach. After a wide experimentation stage, a general goal is to standardise the proposed framework based on the ANSI/ISA-95 international standard as well as to standardise in depth the framework processes. An additional goal is to extend the framework in order to satisfy the needs of service-oriented collaborative networks (SOCN).

Acknowledgements

The work presented in this paper is supported by the project Z-Fact0r which is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 723906. The paper reflects only the authors' views and the Commission is not responsible for any use that may be made of the information it contains.

References

[1] R&D Solutions for Chemicals. Industry 4.0: Top Challenges for Chemical Manufacturing.White Paper. ELSEVIER.

[2] Wang, K.: Intelligent Predictive Maintenance (IPdM) system – Industry 4. 0 scenario, https://www.semanticscholar.org/paper/Intelligent-Predictive-Maintenance-(-IPdM-)-system-Wang; 2015.

[3] Riel, A., Kreiner, C., Macher, G., Messnarz, R.: Integrated design for tackling safety and security challenges of smart products and digital manufacturing. CIRP Ann.; 2017; 66, 177–180.

[4] Mörzinger, B., Weiler, T., Trautner, T., Ayatollahi, I., Angerer, B., Kittl, B.: A large-scale framework for storage, access and analysis of time series data in the manufacturing domain. Procedia CIRP; 2018; 67, 595–600.

[5] Rajkumar, R. (Raj), Lee, I., Sha, L., Stankovic, J.: Cyber-physical systems: The next computing revolution. In: Proceedings of the 47th Design Automation Conference on - DAC '10. ACM Press, New York, USA.; 2010. p. 731.

[6] Andersson, J., Johnson, P.: Architectural integration styles for large-scale enterprise software systems. IEEE Comput. Soc. pp. 224–236.

[7] He, W., Xu, L. Da: Integration of Distributed Enterprise Applications: A Survey. IEEE Trans. Ind. Informatics; 2014; 10, 35–42.

[8] Lihong J., Li Da X., Hongming C., Zuhai J., Fenglin B., Boyi X.: An IoT-Oriented Data Storage Framework in Cloud Computing Platform. IEEE Trans. Ind. Informatics; 2014; 10, 1443–1451.

[9] Razzaque, M.A., Milojevic-Jevric, M., Palade, A., Clarke, S.: Middleware for Internet of Things: A Survey. IEEE Internet Things J.; 2016; 3, 70–95.

[10] Zhou, H.: The internet of things in the cloud : a middleware perspective. CRC Press, Taylor & Francis Group (2013).

[11] Delicato, F.C., Pires, P.F., Batista, T.: Middleware Solutions for the Internet of Things. Springer London, London; (2013).

[12] Roalter, L., Kranz, M., Möller, A.: A middleware for intelligent environments and the internet of things. In: UIC'10 Proceedings of the 7th international conference on Ubiquitous intelligence and computing. 2010. pp. 267–281.

[13] Song, Z., Cardenas, A.A., Masuoka, R.: Semantic middleware for the Internet of Things. Internet of Things (IOT). IEEE. 2010. pp. 1–8.

[14] Hong, Y.: A Resource-Oriented Middleware Framework for Heterogeneous Internet of Things. In: 2012 International Conference on Cloud and Service Computing. IEEE. 2012. pp. 12–16.

[15] T. Teixeira, S. Hachem, V. Issarny, N. Georgantas, Service Oriented Middleware for the Internet of Things: A Perspective, in: Springer, Berlin, Heidelberg, 2011: pp. 220–229.

[16] Perera, C., Jayaraman, P.P., Zaslavsky, A., Georgakopoulos, D., Christen, P.: Mosden: An Internet of Things Middleware for Resource Constrained Mobile Devices. In: 2014 47th Hawaii International Conference on System Sciences. IEEE. 2014. pp. 1053–1062.

[17] Chryssolouris, G., Mourtzis, D., Papakostas, N., Papachatzakis, Z., Xeromerites, S.: Knowledge Management Paradigms in Selected Manufacturing Case Studies. In: Methods and Tools for Effective Knowledge Life-Cycle-Management. Springer Berlin Heidelberg, Berlin, Heidelberg.; 2008. pp. 521–532.

[18] Murray, S.: Knowledge management in manufacturing. Economist; 2007.

[19] W3C: OWL 2 Web Ontology Language - Document Overview. 2012;

[20] Efthymiou, K., Sipsas, K., Mourtzis, D., Chryssolouris, G.: On knowledge reuse for manufacturing systems design and planning: A semantic technology approach. CIRP J. Manuf. Sci. Technol.; 2015; 8, 1–11.

[21] Milicic, A., Perdikakis, A., Kadiri, S.E., Kiritsis, D.: PLM Ontology Exploitation through Inference and Statistical Analysis A Case Study for LCC. IFAC Proc. Vol.; 2013; 46, 1004–1008.

[22] Kádár, B., Terkaj, W., Sacco, M.: Semantic Virtual Factory supporting interoperable modelling and evaluation of production systems. CIRP Ann.; 2013; 62, 443–446.

[23] Wilkinson, K., Sayers, C., Kuno, H.A., Reynolds, D.: Efficient RDF Storage and Retrieval in Jena2. Swdb; 2003; 131–150.

[24] A. Pagoropoulos, J.A.B. Andersen, L.L. Kjær, A. Maier, T.C. McAloone, Building an Ontology of Product/Service-Systems: Using a Maritime Case Study to Elicit Classifications and Characteristics, in: Springer, Berlin, Heidelberg; 2014; pp. 119–126.

[25] Zhang, W., Hansen, K.M.: Semantic Web Based Self-Management for a Pervasive Service Middleware. In: 2008 Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems. IEEE. 2008. pp. 245–254.

[26] Pintzos, G., Matsas, M.: Defining Manufacturing Performance Indicators Using Semantic Ontology Representation. Procedia CIRP; 2012; 3, 8–13.

[27] Botta, A., de Donato, W., Persico, V., Pescape, A.: On the Integration of Cloud Computing and Internet of Things. In: 2014 International Conference on Future Internet of Things and Cloud. IEEE. 2014. pp. 23–30.

[28] Ortiz, A., Lario, F., Ros, L.: Enterprise Integration—Business Processes Integrated Management: a proposal for a methodology to develop Enterprise Integration Programs. Comput. Ind.; 1999; 40, 155–171.

[29] Pahl, C., Hasselbring, W., Voss, M.: Service-centric integration architecture for enterprise software systems. J. Inf. Sci. Eng.; 2009;

[30] Bagheri, B., Lee, J.: Big future for cyber-physical manufacturing systems. In: Design world. 2015.

[31] ISO/IEC: 20922:2016 Information technology -- Message Queuing Telemetry Transport (MQTT) v3.1.1.

[32] Web Services Architecture; https://Www.W3.Org/TR/2004/NOTE-Ws-Arch-20040211/#relwwwrest; 2004.