Micro-Workflows: Kafka and Kepler fusion to support Digital Twins of Industrial Processes

Gleb Radchenko
School of Electrical Engineering and Computer Science
South Ural State University
Chelyabinsk, Russia
gleb.radchenko@susu.ru

Ameer B. A. Alaasam
School of Electrical Engineering and Computer Science
South Ural State University
Chelyabinsk, Russia
alaasamab@susu.ru

Andrey Tchernykh
Computer Science Department
CICESE Research Center
Ensenada, Mexico
South Ural State University
Chelyabinsk, Russia
chernykh@cicese.mx

***Abstract*— In recent years, we observe an exponential growth of "Smart Industry" concept that relies on the use of software and hardware systems to analyze data from several types of smart sensors by various types of models: mathematical, computational, data, etc. A set of such virtual models, representing processes, systems and equipment is called "Digital Twins" (DTs). DTs use data gathered from the sensory systems on production lines to predict failures of machinery, optimize the quality of the products, and reduce the ecological footprint from facilities. They can be described as a sequence of jobs that perform required functionality linked together by a set of edges that represent data dependencies. To organize a flexible cloud computing support for the Digital Twin execution, we propose a concept of Micro-Workflows that combines the power of scientific workflows, the flexibility of containers technology, and robustness of the distributed streaming approach.**

Keywords—digital twins, microservices, containerization, micro-workflows, Kafka, Kepler

I. Introduction

The smart manufacturing approach involves the organization of production process using the intelligent platform that ensures data collection from the sensor networks, data storage and processing, using data mining, and machine learning techniques to improve the efficiency of production [1]. Among the initiatives for the development of these platforms and their introduction into the industry can be identified "Industry 4.0" program [2] developed in Germany, "Smart Manufacturing Leadership Coalition" [3] in the USA, and "Industrial IoT" concept introduced by GE Corporation [4].

One of the key approaches that unite these programs is a *Digital Twin (DT)* concept that supports virtual models of real equipment, industrial process, and final products. It uses methods of analysis of data from diverse types of sensors installed on the objects for tuning and actualization of their virtual state.

The DT uses different mathematical models for the simulation of the processes of interest using statistical methods, data mining, finite element method, etc. [5]. Each of these computing methods defines specific requirements for the necessary computational resources. For example, data mining methods require a high volume of high-throughput storage to aggregate and access the analytical data, and high scalability computing system to process it. On the other hand, a model that uses the finite element method requires a high-performance computing system (or a supercomputer). The only way to fulfill such a variety of requirements to computational resources and provide scalability for simulation is to use cloud computing systems as providers of the computing infrastructure.

To describe the processes on a factory or in sophisticated product lines, the workflow concept is applied. Workflows [6] have emerged in industry, business, and science as an easier way to formalize and structure a process that consists of a set of independent actions connected to each other through dependency relation. A computational workflow provides an efficient solution for complex processes simulation by the orchestration of services and optimization of their execution order [7]. Therefore, to develop a DT of an industrial process or equipment, we can represent it as a computational workflow composed of a set of computational services that represent models for process stages and their interaction.

The distinguishing characteristic of the DT is its capability to capture, transfer, and analyze real-time streaming data from Industrial Internet of Things (IIoT) equipment for tuning and actualization of their virtual state [8]. A solution to these challenges is a streaming data middleware providing modularity, flexibility, and control over complex data interactions [9].

In this paper, we describe a concept of the Digital Twins for industrial process simulation and a model of event-driven approach, combining the power of scientific workflows, the flexibility of containers technology, and robustness of the distributed streaming approach. We provide a reference implementation of this approach by developing new actors to extend Kepler scientific workflow management system to support data stream consumption and production operations using Kafka framework. This approach allows us to develop a workflow as a set of loosely coupled services (Micro-Workflows inspired by Microservices) that deployed inside Docker containers and communicate through streaming middleware.

The paper organized as follows. In Section II, we provide an overview of existing approaches in cloud computing that support the DT concept. In Section III, we describe the DT concept, which allows simulating a real-world manufacturing process. Section IV presents an overview of the concept of the Digital Twins cloud platform. In Section V, we describe the Micro-Workflows approach for the design of Digital Twins. Section VI presents Micro-Workflows implementation and experiment deployment. In Section VII, we provide the results of the Micro-Workflows performance evaluation. In conclusion, a summary of the work and possible directions for further research are presented.

The reported paper was supported by RFBR, research project No. 18-07-01224 and by Act 211 Government of the Russian Federation, contract № 02.A03.21.0011

II. Related Work

Nowadays, one of the most common concepts related to Digital Twins platforms is IoT cloud systems. They are oriented to collect and perform an analysis from various sensors. The authors of the paper [10] describe a "Sensing as a service" (SenaaS) model that consists of four conceptual layers: 1) sensors and sensor owners, 2) sensor publishers, 3) extended service providers, and 4) sensor data consumers. The paper describes several cloud systems for data harvesting from sensors. For instance, Xively [11] is a public cloud for the IoT that simplifies and accelerates the creation, deployment, and management of sensors in a scalable manner. The OpenIoT project [12] focuses on providing an open source middleware framework for the dynamic formulation of self-managed cloud environments for IoT. Global Sensor Networks (GSN) [13] is a middleware, which supports sensor deployment and offers a flexible, zero-programming deployment and integration infrastructure for IoT.

However, developers of such systems emphasize the need for an efficient data collection and analysis. Therefore, the IoT cloud systems could be used for implementation of use-case models of DTs and transfer data to other systems for more precise analysis.

The authors of [14] propose an architecture reference model to design cloud-based cyber-physical systems (C2PS) based on the DT concept. They divide the system into three operational modes: physical level sensors-fusion mode, cyber level digital twin services-fusion mode, and deep integration of sensor-services fusion mode. The authors describe analytically modeled computation, communication, and control properties of the C2PS.

DT can be described as a sequence of jobs that perform required functionality linked together by a set of edges that represent data dependencies between jobs. Such an approach can be implemented as a scientific workflow (SWF) that commonly managed, executed, and monitored by Scientific Workflow Management Systems (SWfMS), such as DiVTB [15], Pegasus [16], Kepler [17], ASKALON [18], or tGSF [19], [20]. SWfMS allows scientists to focus on their research by automating the required experimental scenarios for the complex computational system. However, running one large-scale SWF with tightly-coupled dependencies, sequential execution, and limited streaming support does not meet the event-driven approach that is necessary for the execution of DTs.

III. The Digital Twin

The Digital Twin concept (see Fig. 1) was introduced at the beginning of the 2000s. It is based on an approach of synchronization of the real-world entities with their virtual representation in the form of computational models [21]. The *Digital Twin* is a hierarchical system of mathematical models, computational methods, and software services, which provides near real-time synchronization between the state of the real-world process or system and its virtual copy [8]. Synchronization of mathematical models (multi-physics, multi-scale, and probabilistic) in the Digital Twin with real data allows to measure the efficiency of equipment work, receive information about a state of process and end-products, and predict the indicators using a historical database.

The following characteristics of the DT can be highlighted:

1) *Real-time reflection.* A DT reflects a physical object (equipment, process and system) that keeps ultra-high synchronization and fidelity with the real world.

2) *Hierarchy.* A DT of an industrial process can contain DTs of the process stages, which, in turn, contain DTs of the equipment enforcing these stages.

3) *Self-evolution.* Since a DT provides a reflection of the object or process, it should perform undergoing continuous model improvement by comparing the simulation process with a physical object and process, concurrently [22].

The DT uses diverse types of models to simulate a real object's current state. The comprehensive and well-timed state reflection can be obtained only by usage of models with the accuracy, necessary and sufficient for satisfying the "real-time reflection" requirement. The more accuracy is required, the more resources and time is needed for simulation. Therefore, we can distinguish the following types of models used in DTs.

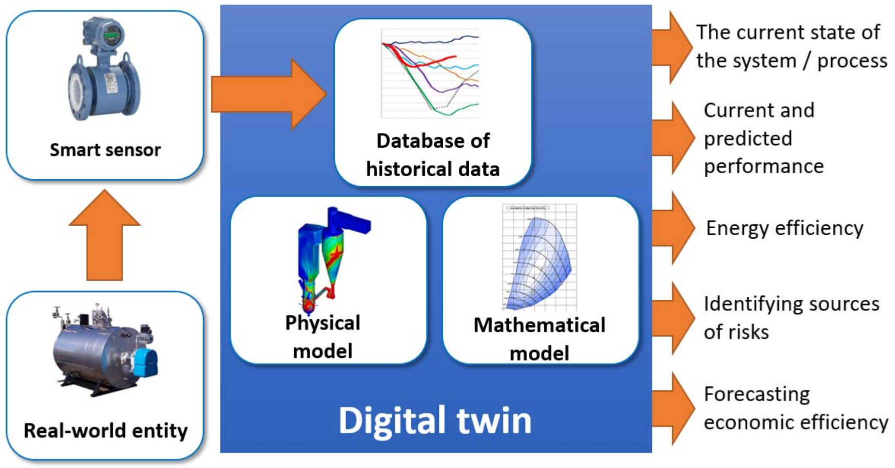
Smart sensor
Real-world entity
Database of historical data
Physical model
Mathematical model
Digital twin
The current state of the system / process
Current and predicted performance
Energy efficiency
Identifying sources of risks
Forecasting economic efficiency

Fig. 1. The Digital Twin concept

– *Case models* are used for up-to-date notification of end users about events when parameters of the technological process exceed the established boundaries. The low computational complexity of such models allows their implementation on low-power control units. At the same time, we need to allocate computing resources to filter out the provided messages and notify the interested executors.

– *Thermodynamic first-principles models* are formed on the basis of thermodynamic, physical and chemical models, underlying the operation of specific equipment. Those models allow forecasting the degradation of the characteristics of technological processes, taking into account the current dynamics, and predict the date and time of the next equipment maintenance considering its actual efficiency [5].

– *Finite element computational models* provide a simulation of complex multifactor processes by solving systems of differential equations with given boundary conditions. They are used in the design of new products, simulation of new technological cycles, and simulation of emergency situations. They are computationally complex problems requiring special computing equipment to execute.

– *Data mining methods* are used to solve problems for which none of the above methods are applicable due to the complexity, multifactorial or stochastic nature of the ongoing processes. These methods require large volumes of good quality (pre-cleaned) data that have different nature of origin.

IV. Digital Twins Cloud Platform

To provide the DT execution, we design a Digital Twins Cloud Platform, which provides a dynamic allocation of computing resources and provides an API to present the DTs as a microservices. Thus, DTs Cloud Platform provides a “Digital Twin-as-a-Service” (DTaaS) cloud model. The DTaaS model presents the DT as a set of cloud services for storage and analysis of data gathered from sensors, simulation of the real-world objects, and visualization of their virtual representation.

The DTs Cloud Platform provides a single entry point for a variety of players in the market of industrial equipment and high-technology products, supporting the following features (see Fig. 2):

– *Manufacturers of the Industrial Equipment* are able to provide consumers with DTs of the supplied equipment and perform remote monitoring and analysis of the use of the supplied equipment to improve the quality of the product;

– *Industrial Enterprises* can build DTs of industrial processes that support analysis and decision making for the development of the enterprise;

– *Research Centers* can develop and promote their analytical models, as well as provide services to support and improve the efficiency of industrial processes and systems.

The DTs Cloud Platform provides the following levels of abstraction:

1. *The level of the Digital Twin user*. On this level, the user can get an access to the available DTs in the form of cloud applications based on the "Software-as-a-Service" model. The user interface of the DT can be presented in various views, like 3D-models, charts, and text messages that describe selected aspects of the simulated real-world entity to specific types of users. For instance, an engineer uses a DT of a gas boiler to detect a burner clogging. The same DT provides a chart of the boiler efficiency depreciation for an economist. Also, the same Digital Twin can be used to predict the date of the maintenance.

2. *The level of the Digital Twin developer.* At this level, the cloud platform provides resources for the development of DTs based on a "Platform-as-a-Service" model. A DT is described as a set of computational workflows, communicating with each other using the message passing interface.

3. *The level of the Computing Service developer.* At this level, the cloud platform provides an API for Computing Service development based on a "Backend-as-a-Service" model. A Computing Service is represented as a microservice

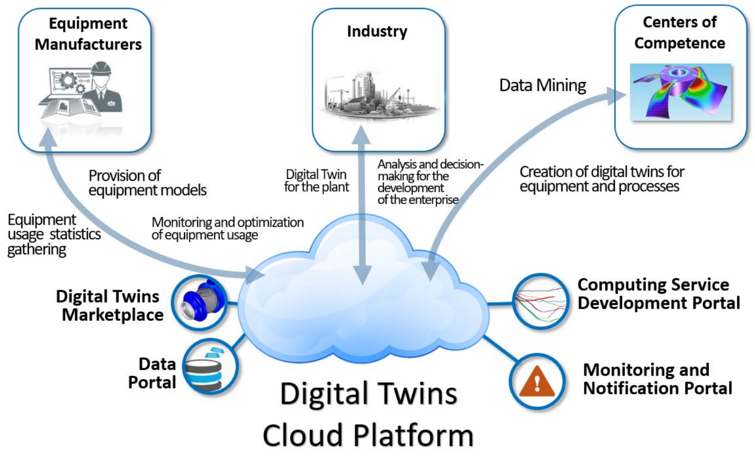
Equipment Manufacturers
Industry
Centers of Competence
Data Mining
Provision of equipment models
Digital Twin for the plant
Analysis and decision-making for the development of the enterprise
Creation of digital twins for equipment and processes
Equipment usage statistics gathering
Monitoring and optimization of equipment usage
Digital Twins Marketplace
Computing Service Development Portal
Data Portal
Monitoring and Notification Portal
Digital Twins Cloud Platform

Fig. 2. Digital Twins Cloud Platform

responsible for specific data processing operation or execution of a specific set of computational methods.

4. *The level of the cloud infrastructure provider.* At this level, instances of Computing Services are mapped to the cloud computing resources provided by the cloud platform based on "Container-as-a-Service" model.

V. Micro-Workflows Approach for the Design of Digital Twins

Current systems that support execution of scientific workflows do not support elastic scalability of sub-workflows and integration of streaming data processing. Such limitations are crucial for the DT implementation.

To address these constraints, we propose to redesign monolithic SWFs into sets of smaller loosely-coupled Micro-Workflows (MWF) that act as independent computational services, deployed in separate containers, which communicate by means of the streaming middleware.

This approach allows IoT low-latency data processing using the MWFs and provides their independent development and deployment. It increases the potential of horizontal scaling and vertical distribution of MWFs. MWFs that should provide low-latency response can be deployed on the cloudlets, located nearby with the IoT equipment, while MWFs that implement batch processing of Big Data can be located on the nodes near storage systems.

To convert the original SWF into a set of MWFs, linked together through the external streaming software, we introduce *Consumer Node* and *Producer Node* to the division points of the original workflow. These nodes allow exchanging the data between MWFs and the streaming middleware. (see Fig. 3).

VI. Micro-Workflows Implementation and Experiment Deployment

To test our approach, we implement the proposed architecture using Kepler[1] as SWfMS, Kafka[2] as a distributed streaming middleware platform that supports horizontal scalability, and Docker as a containerization platform. When loosely coupled services interact using messages, they must provide a generic message format (schema). The Confluent Platform[3] provides the schema registry based on Avro data serialization system[4]. We use docker image provided by Landoop[5] to automate the deployment of Kafka and schema registry in a container.

We evaluate our approach using the sensors data from DEBS[6] 2012 Grand Challenge: Manufacturing equipment. The delay between two consecutive source data points is about 10 ms. We consider the operators of the first query of the challenge. Original data point includes 66 fields. The first set of operators detects the change of state of input fields and emits them along with timestamps of the state change. The second set of operators correlates the change of state of the sensor and the change of state of the valve by calculating the time difference between the occurrence of the state changes and emits them along with timestamps. The goal of the last operators set is to constantly monitor the trend for the calculated values from the previous output data for the period of 24 hours.

To evaluate our approach, we provide several custom actors for Kepler. They extend the capability of Kepler to be able to work with Kafka and process the required computing scenario to meet the first query of the DEBS 2012 workflow, including:

– **KafkaConsumer** actor consumes messages from Kafka source topic, deserializes the received Avro messages, and sends these messages as records to the output port;
– **KafkaProducer** actor receives records from the input port, serializes the received records to the Avro format, and sends them as messages to the Kafka result topic;

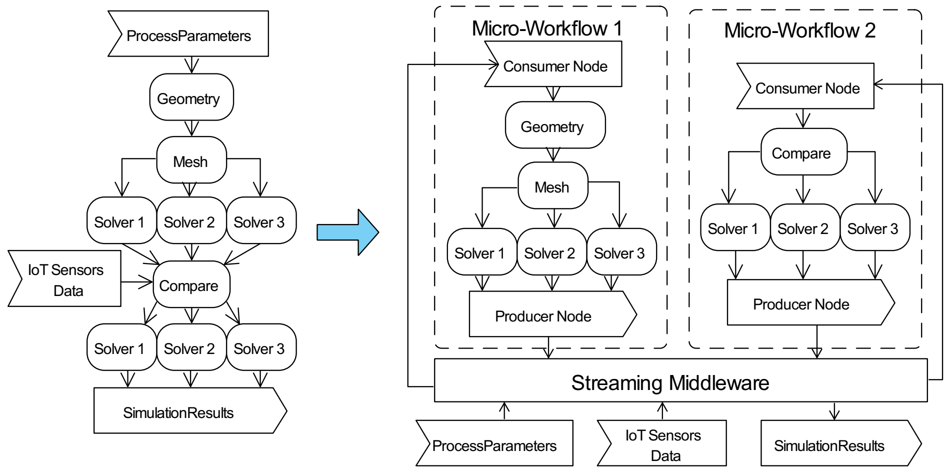

Fig. 3. Micro-Workflows concept to consolidate Workflow and Streaming Middleware

[1] https://kepler-project.org/
[2] https://kafka.apache.org/
[3] https://www.confluent.io/
[4] https://avro.apache.org/
[5] https://www.landoop.com/
[6] http://debs.org/grand-challenges/

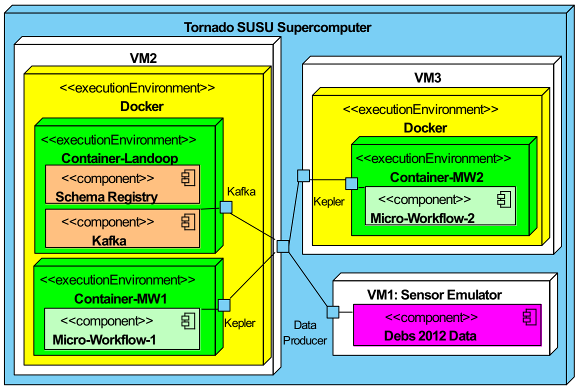

Fig. 4. Deployment of Micro-Workflows on Tornado SUSU Supercomputer

– **DetectStateChange** actor detects a change in two consecutive data records (from 0 to 1 and vice-versa);
– **CorrelateStateChange** actor provides a correlation analysis of the state change in the "0" and "1" time sequences between the couples of sensors. It calculates the time difference between raising (1) and falling (0) edges of each of the input sensor pairs;
– **sXXYY_State** actor extracts correlated time difference values and prepares them for the reflection on the output graph.

The deployment of our solution is provided on the resources of the Tornado Supercomputer at the South Ural State University [23] (Fig. 4). The Sensor Emulator virtual machine (VM1) simulate the process of IoT sensor data generation. It consumes data from DEBS 2012 database and publishes it sequentially into the Kafka, deployed in the Landoop container (VM2). We partition the original DEBS 2012 first query workflow into two Micro-Workflows and pack each Micro-Workflow in a separate container, deployed on VM2 and VM3 virtual machines. Both VM1 and VM3 are provided with 4GB RAM and 4 cores of Intel Xeon X5680 CPU. The VM2 virtual machine is provided with 12 GB RAM and 8 cores of Intel Xeon X5680 CPU.

Micro-Workflow 1 consumes the original sensors data from source Kafka topic and processes them using the **DetectStateChange** actor (see Fig. 5a). The results are published into intermediate Kafka topic. *Micro-Workflow 2* consumes data from the intermediate Kafka topic and processes all the remained set of operators, including plotting of the end-results (see Fig. 5b).

VII. Micro-Workflows Performance Evaluation

To measure the performance of the proposed system, we include the following additional fields to the produced messages:

1- $\boldsymbol{mX_{orgts}}$: time, when the sensor sends the source message number X.
2- $\boldsymbol{mX_{rcvts}}$: time, when Kepler receives the source message X from Kafka source topic.
3- $\boldsymbol{K_{sentts}}$: time, when KafkaProducer actor sends the result message to Kafka result topic.

To evaluate the efficiency of data exchange and processing, we propose to measure the following characteristics:

1- $\boldsymbol{Av_{SM}}$ (average interval between source messages): the average time delay between two consecutive source data messages sent by the sensor to Kafka.
2- $\boldsymbol{Av_{TAT}}$ (average turnaround time**)**: the average time interval required to produce a single result message by our MWF. Note that the interval starts from the time of receiving the second required source message ($\boldsymbol{m2_{rcvts}}$), which participated in any state change from Kafka source, and ends at the time of sending the result message ($\boldsymbol{K_{sentts}}$) to Kafka.
3- $\boldsymbol{Av_{DV}}$ (average delivery time): the average time interval between $\boldsymbol{mX_{orgts}}$ and $\boldsymbol{mX_{rcvts}}$, including data serialization, transferring of the message from VM1 to VM2, acknowledgment of reception of the message in Kafka.
4- $\boldsymbol{Av_{L12}}$ (average latency): the average latency between VM1 and VM2.

Results of our experiments are provided in Table 1. During the 24-hours test, our system processed more than 7.3 million messages with the average turnaround time of about 3.2 ms. The average overhead, introduced to our system by the Kafka streaming middleware can be evaluated as follows:

$$Av_{DV} - Av_{L12} = 4.4 - 3.5 = 0.9\ (ms)$$

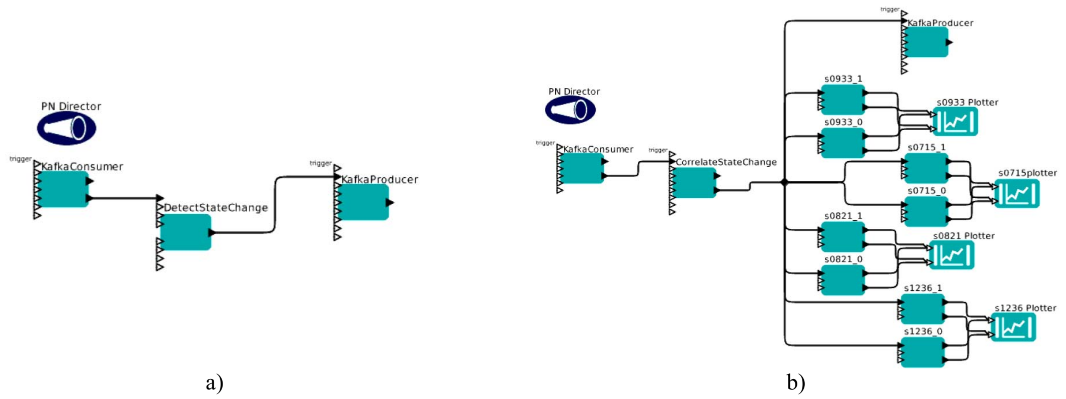

Fig. 5. Implementation of Micro-Workflows for DEBS 2012 data processing

Table 1. Performance evaluation results

Test time	24 hours
Number of messages	7 328 844
Av_{SM}	11.8 ms
Av_{TAT}	3.2 ms
Av_{DV}	4.4 ms
Av_{L12}	3.5 ms

VIII. Conclusion

In this paper, we introduce the concept of the Digital Twins Cloud Platform and propose the Micro-Workflows approach for the implementation of DTs. Our solution extends other existing approaches by combining the power of scientific workflows, the flexibility of containers technology, and robustness of the distributed streaming approach, supporting processing of streaming events from various sources (such as IoT sensors) inside computational workflows. The implementation and testing of this architecture using Kepler SWfMS and Kafka streaming processing solution show that the proposed architecture successfully provides a capability toward meeting the Digital Twins development processes requirements with a reasonable overhead. The main direction of the future work would be an evaluation of the scalability and applicability of this approach to support significant amounts of IoT data in real time.

References

[1] J. Davis, T. Edgar, J. Porter, J. Bernaden, and M. Sarli, "Smart manufacturing, manufacturing intelligence and demand-dynamic performance," *Comput. Chem. Eng.*, vol. 47, pp. 145–156, 2012.

[2] J. Lee, B. Bagheri, and H. Kao, "A Cyber-Physical Systems architecture for Industry 4 . 0-based manufacturing systems," *Manufacturing Letters*, vol. 3. pp. 18–23, 2015.

[3] M. Bryner, "Smart manufacturing: The next revolution," *Chem. Eng. Prog.*, vol. 108, no. 10, pp. 4–12, 2012.

[4] S. Li, L. Da Xu, and S. Zhao, "The internet of things: a survey," *Inf. Syst. Front.*, vol. 17, no. 2, pp. 243–259, 2015.

[5] P. Korambath, J. Wang, A. Kumar, J. Davis, R. Graybill, B. Schott, and M. Baldea, "A smart manufacturing use case: Furnace temperature balancing in steam methane reforming process via kepler workflows," *Procedia Comput. Sci.*, vol. 80, pp. 680–689, 2016.

[6] I. Taylor, E. Deelman, D. Gannon, and M. S. Shields, *Workflows for e-Science*. London: Springer London, 2007.

[7] M. Rahman, R. Hassan, R. Ranjan, and R. Buyya, "Adaptive workflow scheduling for dynamic grid and cloud computing environment," in *Concurrency Computation Practice and Experience*, 2013, vol. 25, no. 13, pp. 1816–1842.

[8] K. Borodulin, G. Radchenko, A. Shestakov, L. Sokolinsky, A. Tchernykh, and R. Prodan, "Towards Digital Twins Cloud Platform : Microservices and Computational Workflows to Rule a Smart Factory," *Proc. the10th Int. Conf. Util. Cloud Comput. - UCC '17*, no. December, pp. 209–210, 2017.

[9] O. Carvalho, E. Roloff, and P. O. A. Navaux, "A Distributed Stream Processing based Architecture for IoT Smart Grids Monitoring," in *Companion Proceedings of the10th International Conference on Utility and Cloud Computing - UCC '17 Companion*, 2017, pp. 9–14.

[10] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Sensing as a service model for smart cities supported by Internet of Things," *Trans. Emerg. Telecommun. Technol.*, vol. 25, no. 1, pp. 81–93, Jan. 2014.

[11] N. Sinha, K. E. Pujitha, and J. S. R. Alex, "Xively based sensing and monitoring system for IoT," in *2015 International Conference on Computer Communication and Informatics, ICCCI 2015*, 2015.

[12] J. Soldatos, N. Kefalakis, M. Hauswirth, M. Serrano, J.-P. Calbimonte, M. Riahi, K. Aberer, P. P. Jayaraman, A. Zaslavsky, I. P. Žarko, L. Skorin-Kapov, and R. Herzog, *OpenIoT: Open source internet-of-things in the cloud*, vol. 9001. 2015.

[13] K. Aberer, M. Hauswirth, and A. Salehi, "Infrastructure for data processing in large-scale interconnected sensor networks," in *Proceedings - IEEE International Conference on Mobile Data Management*, 2007.

[14] K. M. Alam and A. El Saddik, "C2PS: A digital twin architecture reference model for the cloud-based cyber-physical systems," *IEEE Access*, vol. 5, no. January, pp. 2050–2062, 2017.

[15] G. Radchenko and E. Hudyakova, "A service-oriented approach of integration of computer-aided engineering systems in distributed computing environments," in *UNICORE Summit 2012, Proceedings*, 2012, vol. 15, pp. 57–66.

[16] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P. J. Maechling, R. Mayani, W. Chen, R. Ferreira da Silva, M. Livny, and K. Wenger, "Pegasus, a workflow management system for science automation," *Futur. Gener. Comput. Syst.*, vol. 46, pp. 17–35, May 2015.

[17] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludascher, and S. Mock, "Kepler: an extensible system for design and execution of scientific workflows," *Sci. Stat. Database Manag. 2004. Proceedings. 16th Int. Conf.*, vol. I, pp. 423–424, 2004.

[18] T. Fahringer, R. Prodan, R. D. R. Duan, F. Nerieri, S. Podlipnig, J. Q. J. Qin, M. Siddiqui, H.-L. T. H.-L. Truong, A. Villazon, and M. Wieczorek, "ASKALON: a Grid application development and computing environment," *6th IEEE/ACM Int. Work. Grid Comput. 2005.*, 2005.

[19] A. Hirales-Carbajal, A. Tchernykh, T. Roblitz, and R. Yahyapour, "A Grid simulation framework to study advance scheduling strategies for complex workflow applications," in *2010 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW)*, 2010, pp. 1–8.

[20] A. Hirales-Carbajal, A. Tchernykh, R. Yahyapour, J. L. González-García, T. Röblitz, and J. M. Ramírez-Alcaraz, "Multiple Workflow Scheduling Strategies with User Run Time Estimates on a Grid," *J. Grid Comput.*, vol. 10, no. 2, pp. 325–346, Jun. 2012.

[21] F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang, and F. Sui, "Digital twin-driven product design, manufacturing and service with big data," *International Journal of Advanced Manufacturing Technology*. pp. 1–14, 2017.

[22] E. J. Tuegel, A. R. Ingraffea, T. G. Eason, and S. M. Spottswood, "Reengineering aircraft structural life prediction using a digital twin," *Int. J. Aerosp. Eng.*, 2011.

[23] P. S. Kostenetskiy and A. Y. Safonov, "SUSU Supercomputer Resources," in *Proceedings of the 10th Annual International Scientific Conference on Parallel Computing Technologies (PCT 2016). Arkhangelsk, Russia, March 29-31*, 2016, vol. 1576, pp. 561–573.