Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 57 (2016) 213 – 218

www.elsevier.com/locate/procedia

49th CIRP Conference on Manufacturing Systems (CIRP-CMS 2016)

Towards a Rule-Based Manufacturing Integration Assistant

Matthias Wieland[a]*, Pascal Hirmer[a], Frank Steimle[a], Christoph Gröger[a], Bernhard Mitschang[a], Eike Rehder[b], Dominik Lucke[b], Omar Abdul Rahman[b], Thomas Bauernhansl[b]

[a]*Universität Stuttgart, IPVS/AS, Universitätsstraße 38, 70569 Stuttgart, Germany*
[b]*Fraunhofer IPA, Nobelstr. 12, 70569 Stuttgart, Germany*
* Corresponding author. Tel.:+49 711 685 88235; fax: +49 711 685 88424. E-mail address: Matthias.Wieland@informatik.uni-stuttgart.de

Abstract

Recent developments and steadily declining prices in ICT enable an economic application of advanced digital tools in wide areas of manufacturing. Solutions based on concepts and technologies of the "Internet of Things" or "cyber physical systems" can be used to implement monitoring as well as self-organization of production, maintenance or logistics processes. However, integration of new digital tools in existing heterogeneous manufacturing IT systems and integration of machines and devices into manufacturing environments is an expensive and tedious task. Therefore, integration issues on IT and manufacturing level significantly prevent agile manufacturing. Especially small and medium-sized enterprises do not have the expertise or the investment possibilities to realize such an integration. To tackle this issue, we present the approach of the Manufacturing Integration Assistant - MIALinx. The objective is to develop and implement a lightweight and easy-to-use integration solution for small and medium-sized enterprises based on recent web automation technologies. MIALinx aims to simplify the integration using simple programmable, flexible and reusable "IF-THEN" rules that connect occurring situations in manufacturing, such as a machine break down, with corresponding actions, e.g., an automatic maintenance order generation. For this purpose, MIALinx connects sensors and actuators based on defined rules whereas the rule set is defined in a domain-specific, easy-to-use manner to enable rule modeling by domain experts. Through the definition of rule sets, the workers' knowledge can be also externalized. Using manufacturing-approved cloud computing technologies, we enable robustness, security, and a low-effort, low-cost integration of MIALinx into existing manufacturing environments to provide advanced digital tools also for small and medium-sized enterprises.

Peer-review under responsibility of the scientific committee of the 49th CIRP Conference on Manufacturing Systems

Keywords: Rules; Integration; Manufacturing; Smart-Factory; Industrie 4.0

1. Introduction

Today, the use of cyber-physical systems as well as the increasing connectivity of technical devices and machines provide the basis for novel forms of manufacturing operations, especially for self-organization and self-optimization. These aspects are typically subsumed under the term "Industrie 4.0" (I4.0). At this, the flexible linking and integration of heterogeneous manufacturing IT systems and devices, e.g., machines and ERP systems, constitutes a central success factor. Existing integration solutions, such as, workflow systems and manufacturing services busses, represent complex and centrally organized systems, which are costly to adapt and maintain. Thus, they prohibit flexible integration of IT systems and self-organization of manufacturing processes, especially in small and medium-sized manufacturing companies with limited IT budget and IT knowhow.

To address this issue, we present the concept of the Manufacturing Integration Assistant (MIALinx), discuss its implementation in a cloud environment and evaluate its benefits with respect to a real life application scenario in a SME. MIALinx is a lightweight and easy-to-use IT integration solution oriented towards recent web automation platforms and enables rule-based self-organization for manufacturing processes. It supports manufacturers to organize and manage their manufacturing processes using simple, reusable "IF-THEN" rules that link occurring situations in manufacturing environments with corresponding actions.

Peer-review under responsibility of the scientific committee of the 49th CIRP Conference on Manufacturing Systems
doi:10.1016/j.procir.2016.11.037

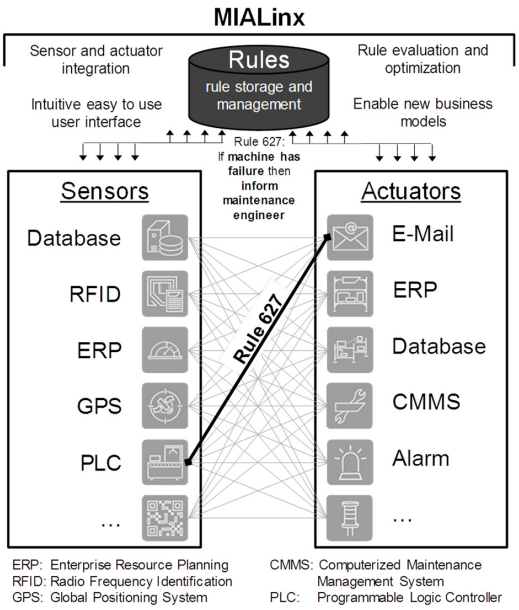

Fig. 1: Concept of MIALinx

Fig. 1 shows the main concept of MIALinx: Rules connect data sources (sensors) and sinks (actuators) based with conditions to model when and if a rule has to be triggered. The rules are stored and managed in a rule database, optimized and rated based on their practical use. Furthermore, MIALinx aims to provide an intuitive user interface for easy modeling of rules.

Fig. 1 shows an example for a rule. A certain machine condition can be defined as a machining break down situation that triggers a notification by sending an e-mail to the responsible maintenance worker. Using this rule, the maintenance worker is automatically informed of any machine failure on the shop floor. Furthermore, it is possible to perform the planning of the maintenance activities in the computerized maintenance management system (CMMS) automatically.

The remainder of the paper is organized as follows: Section 2 details on related work. Section 3 presents the main contribution of the paper and introduces the MIALinx architecture and its components. In Section 4, the two main use-case scenarios are explained – production data acquisition and maintenance. Section 5 gives a summary of the paper and describes future work.

2. Related Work

Existing manufacturing IT environments are coined by a huge heterogeneity of systems and interfaces, ranging from multiple ERP and MES systems of different vendors to various different machine interfaces even in one single factory. To enable data exchange and integration between these systems and interfaces, complex and proprietary point-to-point bridges are typically built in practice [1]. The continuous adaptation of manufacturing processes then leads to significant modification and reimplementation efforts.

A first step towards more simple system integration represent integration middleware approaches, especially the service-oriented architecture (SOA) paradigm using service buses, like the enterprise service bus (ESB) [2], or the manufacturing service bus (MSB) which was especially designed to meet the needs of manufacturing [3]. At this, the Virtual Fort Knox (VFK) approach and implementation deals with running a MSB in a secure cloud environment [4]. In MIALinx, the VFK will be used as to connect to all sensors and actuators available in the production environment with our system. In the SOA context, also workflow technology has become an important tool to connect services. Workflows are commonly used to define and execute business processes. Established standards to define and execute workflows are the Business Process Execution Language [5] (BPEL) and the Business Process Management Notation [6] (BPMN). These standards enable modeling processes and their automatic execution. Workflows ease the integration of different systems, too, but still require strong programming skills to do so. Moreover, they focus on business processes as opposed to manufacturing processes and are thus rarely used on the factory shop floor [7].

Recently, web-automation-platforms like IFTTT [8], or Zapier [9] came up to address easy, simple and flexible system integration. Using these platforms users are able to integrate different IT systems without any programming skills. Nevertheless, this advantage is also one of their biggest drawbacks: to keep up this level of simplicity, these systems only allow users to define rules using simple and predefined conjunctions. Some existing systems, like Zapier, are able to integrate, e.g., Customer Relationship Management Systems. However, they are not able to integrate machines, sensors, or actuators on the factory shop floor as they do not have standardized interfaces. Another disadvantage is that rules are defined independent from each other. For an enterprise system it is necessary to harmonize all rules created by employees and to optimize them. Rule Engines like Jess [10] or Drools [11] are able to do this, but are not designed to operate in an I4.0 environment as they lack manufacturing-specific meta models and interfaces. As we want to use rule engines, another advantage of our approach in contrast to web-automation-platforms is that rule engines support rule definition languages like the Rule Interchange Format (RIF). Using RIF one can easily design a set of rules and deploy them on a rule engine, or export a set of rules from a rule engine and import them in another one.

Some rule engines, e.g., Drools, support also the execution of Complex Event Processing (CEP) queries on a stream of (sensor) data. To cope with a huge number of sensors sending data, a middleware has been developed which is able to distribute the evaluation of CEP queries [12].

There are also other projects who deal with simplifying the modeling of rules: the UC4 Decision System uses a web-based modeling approach similar to the mentioned web-automation-platforms for business users, whereas power users model rules using visual decision graphs and a script language [13]. There is also an approach to use BPMN as a modeling

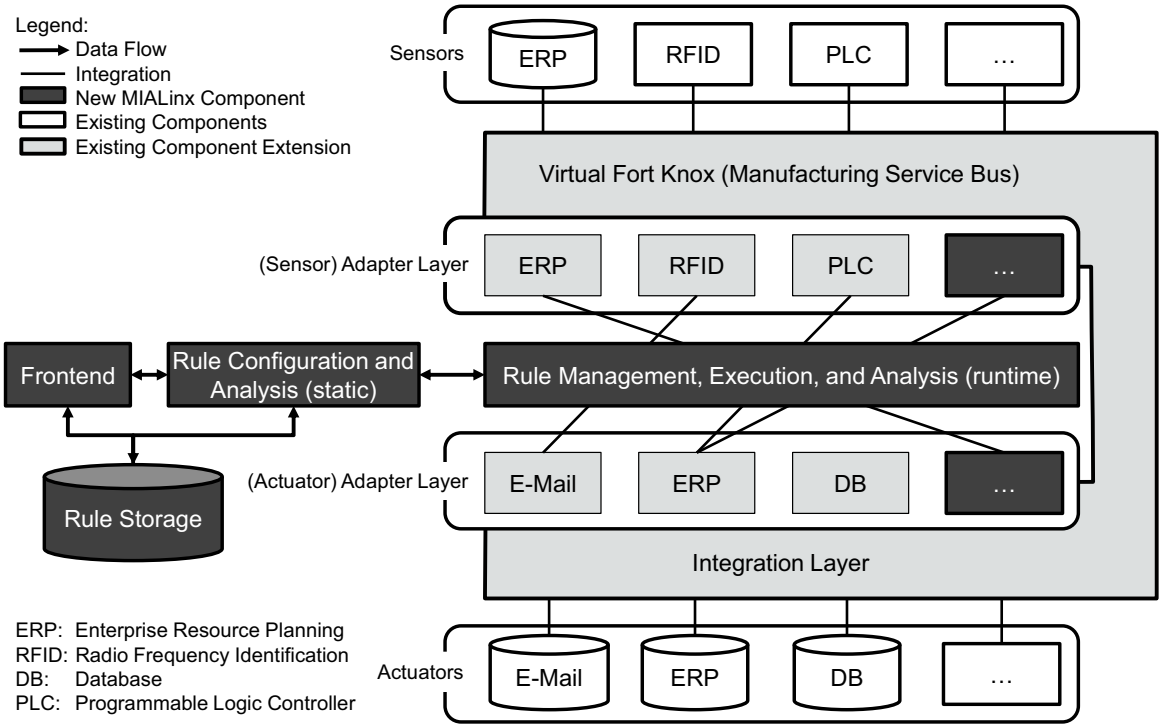

Fig. 2: MIALinx System Architecture

language for business rules [14]. A business rule is the representation of a executable rule in a natural-language-like style [15]. The interested reader is referred to [16] for an overview over existing types of executable rules.

2.1. Preliminary Work

In the area of I4.0 and Internet of Things, a huge number of concepts and tools for sensor modeling and integration have been proposed. These concepts and tools could be used in the MIALinx project as basis for research and implementation. For example, SensorML [17] is an extensible XML-based language for modeling sensors and their characteristics. Whereas SensorML enables modeling of sensors, which produce atomic sensor data. There are approaches that allow users to model aggregated sensor events: SitOPT, developed by IPVS/AS Stuttgart, enables modeling and execution of complex events using so-called Situation Templates [18]. These Situation Templates are complex decision trees and are aggregating low level data like sensor data to complex events, which could be used in MIALinx as input values for the specified rules. Furthermore, in MIALinx sensor-platforms like OpenIoT [19], Global Sensor Network (GSN) [20,21], OpenMTC [22] or FIWARE [23] could be used to integrate sensors and actuators. For providing sensor data M2M standards like ETSI M2M [24], which is supported by, e.g., FIWARE and OpenMTC, or OPC-UA [25] could be used.

3. MIALinx Architecture

Fig. 2 depicts the overall system architecture of the Manufacturing Integration Assistant. The architecture consists of (i) an easy-to-use high comfort *frontend* for usage by domain users, (ii) a *rule storage* to store and manage the IF-THEN rules, (iii) a *rule configuration and analysis* component, which configures the rules, initiates rule execution and conducts static analysis based on the stored rules, (iv) the *rule management and execution* component to process rule execution as well as the initiation of corresponding actions, and (v) a dynamic *rule analysis* component to conduct various analysis steps at runtime. Furthermore, several already existing components are being used, adapted or extended to suite the MIALinx approach (cf. legend in Fig. 2). The MIALinx components are described in detail in the following.

3.1. MIALinx Frontend and Rule Storage

As described in Section 1, one of the main goals of the Manufacturing Integration Assistant is an easy-to-use frontend for modeling of IF-THEN rules that offers an abstraction from technical details to enable usage by domain-experts of SMEs. These rules are usually defined in a machine-readable, technical format such as XML or JSON. By offering an abstraction of these technical structures, e.g., using a graphical modeling interface, rule creation by a wide range of users from different domains can be enabled and

hiring expensive IT experts can be reduced. As mentioned in the previous sections, the rules are modeled in a simple, comprehendible IF-THEN format. The frontend has to support the following functionalities: (i) viewing the modeled rules in a clear, well-understandable way, (ii) rating the rules based on different aspects, e.g., their applicability, to share them amongst other users, (iii) modeling new IF-THEN rules in a simple manner, e.g., by graphically connecting conditions (If-Part) to corresponding actions (Then-Part), and (iv) management of the rules, i.e., deleting, copying, or editing.

The IF-THEN rules are stored in its corresponding format in a *rule storage*, i.e., a data store (e.g., a relational database) that is suitable for the corresponding data structure, optimally offering querying capabilities for an efficient rule retrieval as well as data persistence.

3.2. Rule Configuration and Static Rule Analysis

The main purpose of the *rule configuration and rule analysis* component is the configuration of existing rules, the initiation of the rule deployment, the execution in the corresponding rule engine, e.g., Drools [11], and conducting static analysis based on the created rules. The functionality for rule configuration is accessible through the user interface and enables creating, editing or deleting existing rules stored in the rule storage. As a consequence, this component encapsulates the functionality of the frontend to clearly separate the presentation and the logic layer.

Based on the created rules stored in the rule storage, the *rule analysis* of this component offers a means to efficiently conduct static analysis, e.g., to find conflicting rules or to find rules with the same If-Part that can be combined to a single one, which saves storage space, increases execution efficiency, and keeps the frontend clear. Furthermore, metrics such as “Top 10 rules” can be conducted to further improve the usability of the Manufacturing Integration Assistant. The analysis results are depicted to the users in the frontend.

The rule deployment and execution in a corresponding engine is also initiated through the user interface. On initiation, the rule configuration and analysis component deploys the rule in the corresponding execution engine for processing. If a rule is deleted in the frontend, the rule configuration undeploys it from the engine so it is no longer processed and, furthermore, deletes it in the data storage.

3.3. Adapter Layers

On the *adapter layers*, MIALinx connects to sensors and actuators via the MSB using a single interface. The implemented adapters in this layer describe, process, and convert data from sensors and actuators to the MIALinx rule execution component.

Through the adapter layer, interfaces of existing components such as ERP systems, GPS sensors, Exchange Servers or even documents can be accessed. The goal of this adapter layer is high extensibility as well as offering a means to deploy new adapters, i.e. interfaces to sensors and actuators, in a low-effort manner. To realize this, a large amount of established adapter platforms have been created such as OpenMTC [22] or FIWARE [23]. In previous work [18], we introduced an approach to provision sensor data as uniform REST resources by coping with the sensor integration using tailor-made sensor adapters. In addition, in recent years, many other sensor and actuator binding approaches and platforms have been developed [19,20]. Consequently, there is no need for a new solution; we can build on existing platforms to realize the adapter layer.

The most important aspect is that the sensor and actuator adapter layers have to be accessible by the rule execution engine to (i) receive sensor values for rule evaluation and to (ii) initiate the actuators to react accordingly (e.g., by sending an email). Consequently, a corresponding interface that is able to provide sensor data and initiate actuators has to be provided by this layer.

3.4. Integration Layer / Manufacturing Service Bus

The integration layer connects hard- and software sensors and actuators to MIALinx. The sensors are used to evaluate the condition part of the rules (If-Part) and range from low-level physical sensors to sophisticated enterprise applications such as ERP systems. Similarly, actuators are also from a wide range of devices or applications. This heterogeneity leads to many challenges for the creation of a generic layer.

The integration layer of MIALinx will be built by means of a Manufacturing Service Bus (MSB). The MSB is used to connect MIALinx to a wide range of sensors and actuators. Connecting hard- and software components to MIALinx via MSB brings several advantages compared to a direct interface: the MSB is designed to operate with thousands of interfaces and handles any common communication protocol and therefore enables high scalability of the system. Additionally, it brings a sophisticated message queue, which allows the processing of almost unlimited rule executions. Deploying the MSB in a cloud computing environment, computing power can be easily aligned with the demand.

The manufacturing cloud platform Virtual Fort Knox [4] provides the necessary secure, scalable and robust basis for the deployment of MIALinx.

3.5. Rule Management and Execution

The rule management and execution component represents the main component of the Manufacturing Integration Assistant. This component consists of a rule execution engine, e.g. using Complex Event Processing such as Drools, and a management component that provides deploying, undeploying starting and stopping the rule evaluation, as well as management of the rules. Furthermore, this component is connected to the sensor and actuator adapter layers as well as to the integration layer, i.e. the manufacturing service bus, to communicate with external physical sensors, actuators, and applications. This interconnection is needed for receiving sensor values to be evaluated by the rules and by initiating actions of the actuators once a rule is evaluated to true.

In some cases, only one actuator is initiated, however, oftentimes it makes sense to initiate several actuators once a condition matches. For example, when ordering material

through an ERP system and additionally writing an email to the responsible person to notify that the order has been placed. When initiating more than one action, the order of their execution can be very important. In the example, it could be necessary to send out the email only if the order has been successfully placed in the system. To realize this, we need a more sophisticated means, e.g., using workflow technologies. By creating so called *action processes*, it is possible to define the order of the actions to be executed, respectively, if some of the actions can be processed in parallel. We decided to only use a simple subset of approved workflow languages (e.g., only using BPEL [5] assign and invoke) to enable a fail-safe execution. The single steps of the workflow can then use the interfaces of the actuator adapter layer and of the integration layer to initiate the corresponding actions.

3.6. Rule Analysis at Runtime

Aside the evaluation of rules that connect sensors and actuators, MIALinx furthermore offers functionalities to conduct analysis at runtime based on the data incurring during rule execution. By doing so, e.g., new rules can be proposed based on the currently deployed ones. The detailed characteristics of this analysis is part of our future work.

4. Planned Scenarios for Evaluation and Validation in Small and Medium Enterprises

Two initial scenarios are planned to implement and evaluate the envisioned architecture: production data acquisition and maintenance.

4.1. Production Data Acquisition Scenario

The application scenario of production data acquisition aims primarily to the dynamic and situational collection and processing of current data in production. These data mainly include status messages of production equipment (e.g., condition of a machine) and employees (e.g., processing progress of a work order).

In practice, SMEs show that the collection of operational data is often associated with great effort and therefore the potential is not fully exploited yet. A distinction can primarily be made between the manual collection (paper-based or by means of legacy IT systems) whose disadvantage is mainly the effort in preparation and processing. This compares with the machine or semi-automatic data acquisition (e.g., by means of RFID), of which preparation and processing can be automated. The disadvantages of these solutions are particularly relevant for SMEs: Investment costs and supporting efforts are a hurdle for many SMEs to implement factory data acquisition tools. Furthermore, once introduced solutions are relatively rigid in their functionality, adjustments to changing conditions are complicated and costly.

The production data acquisition scenario of MIALinx aims to combine the advantages of both solutions (manual data collection: flexibility and costs; automated data collection: processing effort, low time delay). In this context, the application scenario will include solutions for SMEs in two aspects: The data acquisition will be simplified for the user by means of sensor applications, which can be customized depending on the situation and the users preferences. The second aspect addresses the automatic processing of the collected data. Only in few cases (e.g., for legal documentation requirements) the storage of data itself serves a purpose. In most cases, only the triggering of actions based on the collected data brings real benefit for manufacturing companies. In the application scenario of MIALinx, it will firstly be possible to derive events from patterns and trends (e.g., the exceeding of a critical quality parameter). On the other hand, discrete events or indirectly generated events can also be used to trigger appropriate actions, for example to inform responsible workers or the automatic parameterization of machines. Examples of possible rules are:

- **IF:** quality of process step is not in required range
  THEN: generate a quality report in a database.
- **IF:** a production order is not completed in time
  THEN: notify the responsible sales employee by e-mail
- **IF:** the stock of a material gets too low
  THEN: order replenishment in the ERP system
- **IF:** a manufacturing process is almost finished
  THEN: inform the logistics staff on the smartwatch to organize the transportation.

In summary, the MIALinx application scenario of production data acquisition will be a flexible and comprehensive solution to collect, transform and process production data for SMEs.

4.2. Maintenance Scenario

Maintenance is an extremely challenging field for application of digital tools. The planning horizon of maintenance tasks varies extremely. It ranges from term planned tasks often occurring periodically such as inspection tasks to machine failures, requiring an immediate reaction. Objectives are to maximize the availability of a production system with minimized effort and cost. For a partly support of the tasks of course digital tools, such as a computerized maintenance management system (CMMS), are already heavily in use for maintenance planning, work execution, documentation and spare part management. CMMS are often part of an ERP system. Condition monitoring systems and models predicting the remaining useful lifetime are used to monitor the condition of highly critical components in order to avoid unplanned failures. These digital tools here already provide a wide range of information exchange interfaces, but the integration requires huge efforts and is still not flexible enough for maintenance demands. The envisioned architecture provides a solution to simplify this challenge. The maintenance planner and worker can easily program rules himself and connect different required sensors and actuators. Examples of rules for maintenance workers and planners are:

- **IF**: the remaining lifetime of a component is less than 10% or 20h
  THEN: create a maintenance order in the CMMS System.
- **IF**: the remaining lifetime of a component is less than 10% or 20h
  THEN: switch the signal lamp to yellow.

- **IF**: a monitored component fails
  THEN: send a message to the smart watch of the maintenance worker.
- **IF**: If an ordered spare part arrives at the warehouse
  THEN: send an email to the maintenance worker.

5. Conclusion and Future Work

In this paper, we present an approach for realizing a rule-based Manufacturing Integration Assistant (MIALinx). Two main usage scenarios for such an assistant are described and form a promising benefit for SMEs to improve the production on the shop floor in the area of I4.0. For this purpose, MIALinx connects sensors and actuators based on defined rules whereas domain experts, e.g., production workers, without extensive IT knowledge, define the rule-set in a domain-specific, easy-to-use manner. Furthermore, we present the architecture of MIALinx, discuss its implementation in a cloud environment and evaluate its benefits with respect to concrete application scenarios.

In summary, MIALinx improves the flexibility of manufacturing integration both on the IT level and in manufacturing environments by supporting the rule-based integration of various IT systems and shop floor artifacts like machines, sensors or actuators.

As future work, we plan to implement the MIALinx system and develop advanced methods that allow an easy modeling of rules, determine the needed complexity of the rules, and optimize and evaluate the rules based on their practical use.

Acknowledgements

MIALinx is a future research and implementation project commissioned by the Baden-Württemberg Stiftung gGmbH.

References

[1] Kletti J. Manufacturing Execution Systems (MES). Berlin; London: Springer; 2007.
[2] Chappell DA. Enterprise Service Bus. 1st ed. Sebastopol, Calif: O'Reilly; 2004.
[3] Minguez J, Lucke D, Jakob M, Constantinescu C, Mitschang B. Introducing SOA into Production Environments - The Manufacturing Service Bus. Proc. 43rd CIRP Int. Conf. Manuf. Syst., Vienna, Graz, Austria: 2010, p. 1117–24.
[4] Holtewert P, Wutzke R, Seidelmann J, Bauernhansl T. Virtual Fort Knox Federative, Secure and Cloud-based Platform for Manufacturing. Procedia CIRP 2013;7:527–32. doi:10.1016/j.procir.2013.06.027.
[5] Web Services Business Process Execution Language (WS-BPEL) Version 2.0. OASIS; 2007.
[6] Business Process Model and Notation (BPMN) Version 2.0. Object Management Group; 2011.
[7] Gröger C, Niedermann F, Schwarz H, Mitschang B. Supporting Manufacturing Design by Analytics, Continuous Collaborative Process Improvement Enabled by the Advanced Manufacturing Analytics Platform. 2012 IEEE 16th Int. Conf. Comput. Support. Coop. Work Des. CSCWD, 2012, p. 793–9. doi:10.1109/CSCWD.2012.6221911.
[8] IFTTT - Make Your Work Flow n.d. https://ifttt.com/ (accessed December 18, 2015).
[9] The best apps. Better together. - Zapier n.d. https://zapier.com/ (accessed December 18, 2015).
[10] Jess, the Rule Engine for the Java Platform n.d. http://jessrules.com/ (accessed December 18, 2015).
[11] Drools - Business Rules Management System (Java™, Open Source) n.d. http://www.drools.org/ (accessed December 18, 2015).
[12] Cipriani N, Nicklas D, Grossmann M, Hönle N, Lübbe C, Mitschang B. Verteilte Datenstromverarbeitung von Sensordaten. Datenbank-Spektrum 2009;9:37–43.
[13] Obweger H, Schiefer J, Suntinger M, Kepplinger P, Rozsnyai S. User-oriented Rule Management for Event-based Applications. Proc. 5th ACM Int. Conf. Distrib. Event-Based Syst., New York, NY, USA: ACM; 2011, p. 39–48. doi:10.1145/2002259.2002266.
[14] Di Bona D, Lo Re G, Aiello G, Tamburo A, Alessi M. A Methodology for Graphical Modeling of Business Rules. 2011 Fifth UKSim Eur. Symp. Comput. Model. Simul. EMS, 2011, p. 102–6. doi:10.1109/EMS.2011.68.
[15] Hay D. Defining Business Rules – What Are They Really. Final Report. 2000.
[16] Paschke A, Kozlenkov A. Rule-Based Event Processing and Reaction Rules. In: Governatori G, Hall J, Paschke A, editors. Rule Interchange Appl., Springer Berlin Heidelberg; 2009, p. 53–66.
[17] Open Geospatial Consortium I. Sensor Model Language (SensorML) 2007.
[18] Hirmer P, Wieland M, Schwarz H, Mitschang B, Breitenbücher U, Leymann F. SitRS - A Situation Recognition Service based on Modeling and Executing Situation Templates. In: Barzen J, Khalaf R, Leymann F, Mitschang B, editors. Proc. 9th Symp. Summer Sch. Serv.-Oriented Comput., vol. RC25564, IBM Research Report; 2015, p. 113–27.
[19] Soldatos J, Kefalakis N, Hauswirth M, Serrano M, Calbimonte J-P, Riahi M, et al. OpenIoT: Open Source Internet-of-Things in the Cloud. In: Žarko IP, Pripužić K, Serrano M, editors. Interoperability Open-Source Solut. Internet Things, Springer International Publishing; 2015, p. 13–25.
[20] Aberer K, Hauswirth M, Salehi A. A Middleware for Fast and Flexible Sensor Network Deployment. Proc. 32Nd Int. Conf. Very Large Data Bases, Seoul, Korea: VLDB Endowment; 2006, p. 1199–202.
[21] Aberer K, Hauswirth M, Salehi A. Invited Talk: Zero-Programming Sensor Network Deployment. Int. Symp. Appl. Internet Workshop 2007 St. Workshop 2007, 2007, p. 1–1. doi:10.1109/SAINT-W.2007.57.
[22] Boosting the Development of Innovative M2M and IoT Applications n.d. http://www.openmtc.org/ (accessed December 18, 2015).
[23] FIWARE n.d. https://www.fiware.org/ (accessed December 18, 2015).
[24] Machine-to-Machine communications (M2M); Functional architecture. ETSI; 2013.
[25] Leitner S-H, Mahnke W. OPC UA–Service-Oriented Architecture for Industrial Applications. ABB Corp Res Cent 2006.