Context-Aware Documentation in the Smart Factory[1]

Ulrich Beez[†], Jürgen Bock[††], Tilman Deuschel[†], Bernhard G. Humm[†], Lukas Kaupp[†]

{ulrich.beez,tilman.deuschel,bernhard.humm,lukas.kaupp}@h-da.de
[†]Hochschule Darmstadt -
University of Applied Sciences
Haardtring 100
64295 Darmstadt, Germany

juergen.bock@kuka.com
[††]KUKA Roboter GmbH
Zugspitzstraße 140
86165 Augsburg, Germany

Keywords: Smart factory, technical documentation, ontology, semantic matching, robotics

Abstract: When machine errors occur in factories, it is important to act quickly in an appropriate way. Depending on the complexity of the error situation and the skill level of the personnel, it is important to identify the appropriate technical documentation quickly. This paper presents a methodology for semantically matching symptoms and causes in error situations, and automatically presenting solutions to end-users in an intuitive way. For demonstration purposes, the use case of robotics application development is chosen.

1. Introduction

Whenever personnel of smart factories are confronted with machine errors, a critical aspect is the time required to put the machine back to operation. A machine error may reduce the overall productivity and hence impact the factories' competitiveness (Zhang and Ordóñez, 2012, p IX; Saalmann and Hellingrath, 2016, p 601).

We illustrate this by means of the example use case of robot application development for the smart factory. Developers program robots to support dedicated factory processes. One feature of modern robots like e.g. KUKA LBR iiwa 14 R820 is its impedance control mode, i.e., motions can be programmed based on measured forces and torques. (KUKA, 2017). A common use case for impedance-controlled motion is the search of the physical workspace, e.g., a desk. The robot moves in one direction until it senses a specific force, i.e., until it reaches contact. Similar to a human elbow, the joints of the LBR iiwa have angle limits (An et al., 1989, p 1251; KUKA, 2017, p 30). In case of an impedance-controlled motion, a joint can reach its maximum angle if the expected force is not detected during the motion, and no default behaviour is programmed, resulting in a sudden stop during movement. To continue the desired movement, the application developer has to first determine which joint has hit its angle limit. Then, he has to rotate the joint out of its maximum angle and use a combination of other joints for moving to the desired position.

In general, error isolation until its root-cause is found can be a complex and time-consuming task (Lettnin and Winterholer, 2017, p 2). The time required depends on a multitude of factors, such as the machine's complexity, the personnel's skill level and the availability of other information sources. Recent government and industry initiatives have focused on optimizing the manufacturing processes by studying on smart factories that use highly interconnected machines. For Germany: Industrie 4.0

[1] The project ProDok 4.0 is funded by the German Federal Ministry of Education and Research (BMBF) within the framework of the Services 2010 action plan under funding no. 02K14A110.

(Henninger et al., 2013), For the USA: The Industrial Internet (Industrial Internet Consortium 2014), For France: Industrie du Futur (2015). More precisely, according to Henninger et al. (2013), those initiatives have the potential to compensate short time deficiencies, allow flexible reaction to errors and even improve the work-life balance. The charts of Bauernhansl et al. (2014, p 31) indicate that the so-called next industrial revolution has the potential to introduce high savings in different industry cost areas. The German industry association for digital business Bitkom (2017) offers a closer insight: One of their recent press releases contains an excerpt of a study conducted in 314 companies with 3 or more employees stating that for Germany there is a possibility for an increase in productivity of up to 78.5 billion (10^9) euros until the year 2025. With regard to the number of ongoing research projects and their topics, a not sufficiently answered question might be how smart factories with increased connectivity can be utilized to achieve such a growth in productivity. One indicator is the web page of the German Federal Ministry of Education and Research (2017) which states in Germany, there are over 300 research project partner locations for ongoing projects within the smart factory domain. Some projects examine holistic approaches, e.g. the recently completed CyberSystemConnector (CSC) project interlinks distributed technical documentation (CyberSystemConnector, 2017). Other projects focus on specific scenarios. E.g. Lin et al. (2012) is introducing a combination of statistics and fuzzy logic for machine failure detection. A study of real time control systems for sensor networks in the same domain is conducted by Nguyen (2017).

In this paper, we propose a novel methodology for reducing the time span for fixing machine errors in smart factories by automatically providing appropriate technical documentation taking machine context into account. In the example use case presented above the robot application developer will automatically be alerted when a robot joint has reached its maximum angle. The information of how to move the robot back into the desired position will be provided in an intuitive way, integrated in the developer's workflow.

This paper is structured with five sections in total. Section 2 describes the problem. In Section 3, the proposed solution is described, which is evaluated in Section 4. A review of related literature can be found in Section 5. The papers' last section comprises conclusions and future work.

2. Problem Description

Based on expert interviews conducted with personnel of manufacturers for smart factory equipment (four robot application developers, one knowledge engineer, three support engineers) the following requirements have been collected:

R1) In case a machine error occurs in a smart factory, personnel shall be enabled to *resolve the error quickly and with little effort*.

R2) *Appropriate technical documentation* shall be provided, indicating causes and solutions of machine errors.

R3) The provided technical documentation shall *match the machine context* in which the error occurred.

R4) The technical documentation shall be provided *automatically*, alerting the user as soon as the machine error occurs.

R5) *Devices* shall be supported which support the smart factory workflow, e.g., desktop computer, tablet PC, smartphone, or smartwatch.

R6) User *interaction* shall be *fast* to not disturb the personnel's workflow.

3. A Method for Context-Aware Documentation in the Smart Factory

3.1. User Interaction Concept

We explain the interaction concept of the methodology by means of the example use case introduced above. As soon as the robot stops during hand guiding, the user is alerted. This alert may be pushed to a suitable device, e.g., the development workstation, a tablet PC or even a smartwatch. The alert indicates the symptom of the machine error, i.e. 'Joint: the robot has stopped'. See Figure 1 for a screenshot of a dashboard view on the development workstation.

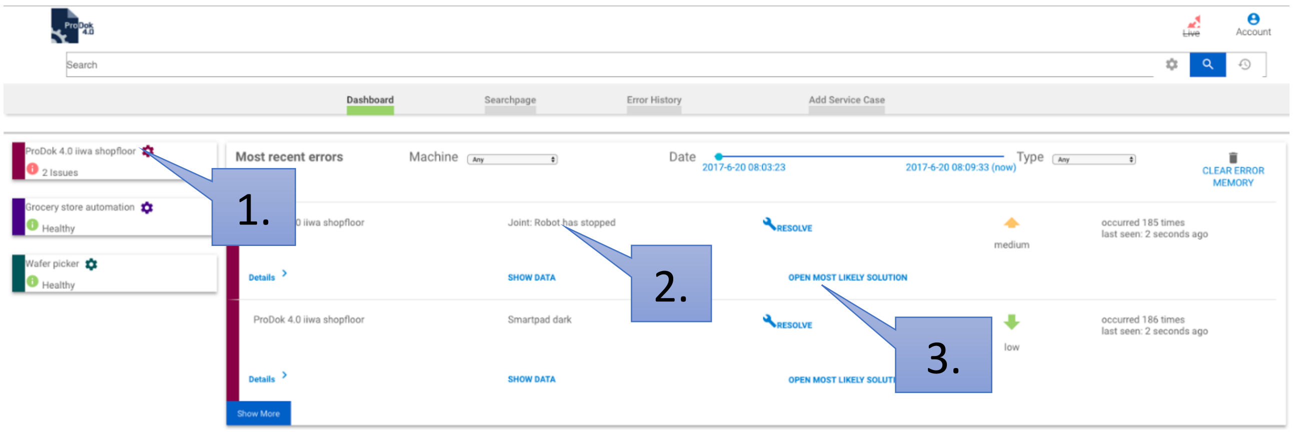


Figure 1: Dashboard View

An important aspect of the screen design is the clarity of information presentation, following the ISO Standard 9241-110 (ISO, 2006), respectively 9241-210 (ISO, 2010). The user centred design approach is applied, which includes close contact to the end user to define requirements collaboratively and test intermediate prototypes iteratively (Garret, 2011).

The dashboard component implements the interaction design pattern of sequence-of-use, which follows the mental model of spatial alignment of semantically related objects (Koffka, 2014). Elements that share a semantic relationship are grouped together and the interaction design provides subsequent interaction-steps for the main tasks.

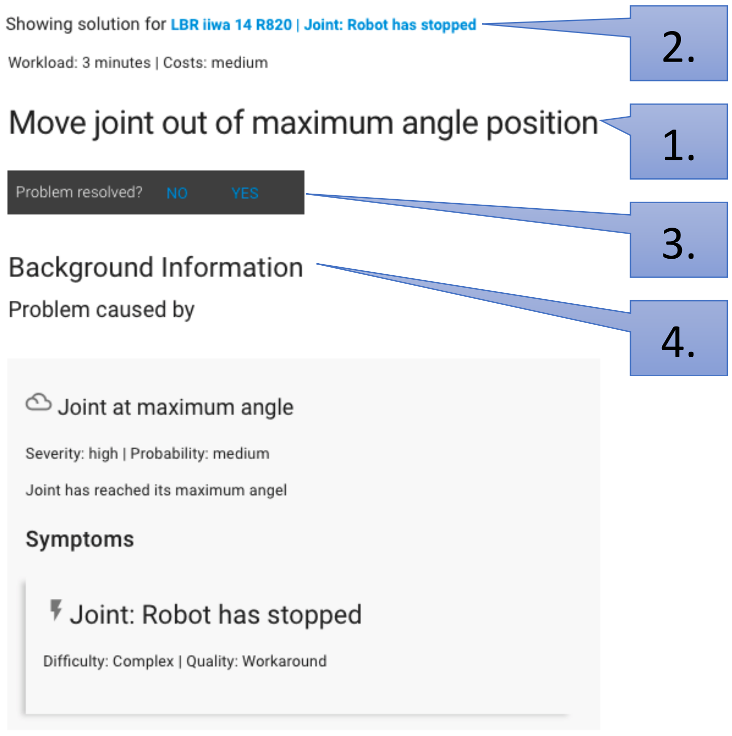
The dashboard's most important components are:

(a) The overview functionality of all connected devices as a list (Figure 1, Mark 1).

(b) A table containing the most recent errors for all connected devices. Each column displays the error symptom alongside a shortcut towards the solution. E.g. 'Joint: the robot has stopped'

(Figure 1, Mark 2) and in addition a navigation component reducing the effort a user has to invest for finding a solution, e.g. 'open most likely solution', (Figure 1, Mark 3)

With a single interaction, e.g., a click on 'open most likely solution' (Figure 1, Mark 3), the user is provided with a solution to the most likely cause of this machine error. See Figure 2.


Showing solution for LBR iiwa 14 R820 | Joint: Robot has stopped

Workload: 3 minutes | Costs: medium

Move joint out of maximum angle position

Problem resolved? NO YES

Background Information

Problem caused by

Joint at maximum angle

Severity: high | Probability: medium

Joint has reached its maximum angel

Symptoms

Joint: Robot has stopped

Difficulty: Complex | Quality: Workaround

Figure 2: Solution View

The conceptual ideas of the Solution View (Figure 2) are as follows:

(a) Present the solution to the machine error, e.g. the solution text 'Move joint out of maximum angle position' (Figure 2, Mark 1).

(b) Provide a quick view to the user regarding error context and symptom, e.g. the blue headline 'LBR iiwa 14 R820 | Joint: the robot has stopped' (Figure 1, Mark 2).

(c) Collect user feedback for a solution, e.g. 'Problem resolved?' (Figure 2, Mark 3).

(d) Display solution background information to the user, matching both error context and error, e.g. the cause and symptom (Figure 2, Mark 4).

(e) In case that several causes for the machine error symptom (Joint: the robot has stopped) exist, they are sorted by likelihood in descending order. On the solution view following the most likely solution, the less likely ones are displayed.

3.2. Software Architecture

The software architecture is shown below in Figure 3 and consists of the 3 *layers* (Starke, 2015): *Presentation*, *Logic* and *Data.* Each layer encompasses different modules. Following Figure 3 the purpose of each module is described.

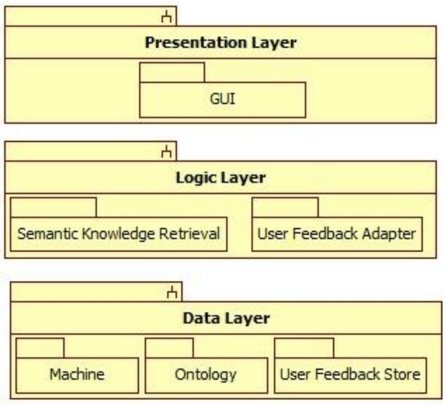


Figure 3: UML Class Diagram – Layer View

Presentation Layer: It contains the Graphical User Interface (*GUI*) which servers as entry point for the user.
Logic Layer: This layer encompasses two modules (a) the *Semantic Knowledge Retrieval*, with the purpose of providing accurately fitting documentation and (b) the *User Feedback Adapter* which has the purpose of handling user feedback.
Data Layer: Three modules are located here, (a) the *Machine*, which sends out event and context information, (b) the *Ontology*, containing hierarchies of products and errors, both interlinked, and modularized technical documentation and (C) the *User Feedback Store*, containing collected user feedback.

For providing an inside view of the components on the different layers, their interaction is shown in Figure 4 which is explained afterwards applying the introductions use case.

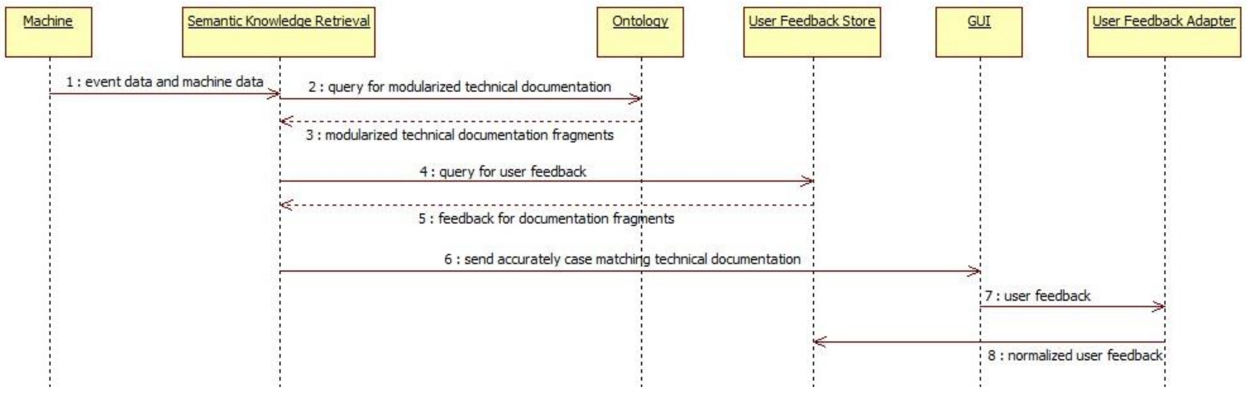

Machine
Semantic Knowledge Retrieval
Ontology
User Feedback Store
GUI
User Feedback Adapter
1 : event data and machine data
2 : query for modularized technical documentation
3 : modularized technical documentation fragments
4 : query for user feedback
5 : feedback for documentation fragments
6 : send accurately case matching technical documentation
7 : user feedback
8 : normalized user feedback

Figure 4: UML Sequence Diagram – Smart Documentation Retrieval

Step 1: As soon as an error occurs, the machine sends *event data* and *machine data* to the *'Semantic Knowledge Retrieval'* component. *Event dat*a describes the machine error in detail, e.g. 'Joint 3 maximum angle reached, emergency stop' on 13:45:17. *Machine data* contains context information, e.g. the robots' digital identification plate with manufacturer and type, e.g.,'KUKA' and 'LBR iiwa 14 R820'.

Step 2,3: Modularized technical documentation is retrieved from the ontology by querying for the event data in combination with the machine data. The modularized technical documentation consists of the documentation fragments *symptom*, *cause,* and *solution (SCS)*.

Step 4,5: User feedback is queried for each of the previously retrieved SCSs.

Step 6: *Technical documentation* accurately matching both the machine event and the personnel's preference is forwarded to the *'Graphical User Interface' (GUI)* component.

Step 7,8: As soon as the GUI sends *user feedback* it is saved in the feedback store after normalization.

3.3. Ontology

An *ontology* specifies concepts and their relationships. One purpose of an ontology is to bridge terminology across different domains (Busse et al., 2015). Here, the event data and machine data, both originating from the machine, are bridged to the technical documentation. The ontology can be queried to retrieve *symptoms*, *causes* and *solutions* (SCS), matching both product and error data. See Figure 4 with an example following the W3C recommendation for concepts and abstract syntax (W3C, 2014).

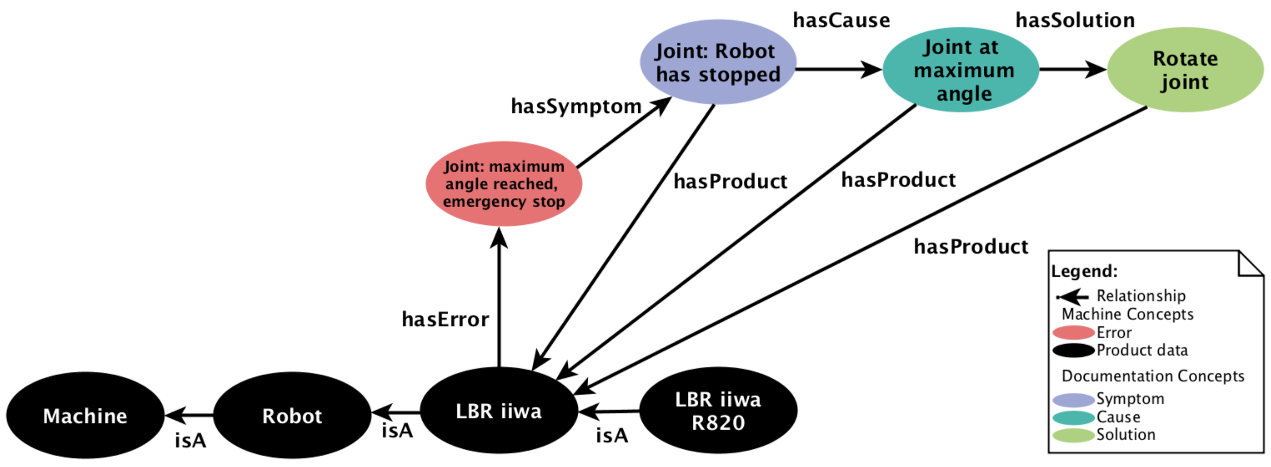


Figure 4: Ontology example

Within the ontology, different concepts are modelled:

(a) Hierarchies of products and errors, both interlinked, e.g.:
'LBR iiwa R820' 'isA' 'LBR iiwa'
'LBR iiwa' 'has Error' Joint: maximum angle reached, emergency stop'

(b) Technical documentation splitted in symptoms, their causes and solutions (SCS) with linkages to (a), e.g.:
‘Joint: maximum angle reached, emergency stop’ ‘hasSymptom’ ‘Joint: Robot has stopped’
‘Joint: Robot has stopped’ ‘hasCause’ ‘Joint at maximum angle’
‘Joint at maximum angle’ ‘hasSolution’ ‘Rotate joint’
all linked to ‘LBR iiwa’ via ‘hasProduct’

The ontology enables modelling transitive relationships like ‘isA’. ‘LBR iiwa R820’ has no direct relationship with any error, symptom, cause, or solution. However, due to the ‘isA’ relationship with ‘LBR iiwa’, the relationships to the respective error, symptom, cause, and solution can be inferred.

In addition to the label attribute shown in Figure 4, concepts and relationships may have additional attributes. E.g., a solution may have an additional attribute containing a detailed description on how to apply the solution. An additional attribute for SCS vertices may contain information about the target user group.

3.4. Semantic Knowledge Retrieval

The aim of the *‘Semantic Knowledge Retrieval’* component is providing appropriate technical documentation. The component is triggered by the machine as soon as an error (event) occurs on the machine. The trigger mechanism hands over event data and machine data. Those data are used to query the ontology for technical documentation (symptoms, causes and solutions - SCS). Subsequently, the component queries the user feedback store to retrieve the feedback for the SCSs in order to rank the SCSs according to user preferences. Finally, the ranked SCSs are forwarded to the GUI. Semantic knowledge retrieval is a most complex process including pre-processing, semantic enrichment, ontology querying, and ranking. A detailed description is beyond the scope of this paper and can be found in Kaupp et al. (2017).

3.5. User Feedback Adapter

The *‘User Feedback Adapter’* is tracking interactions and feedback originating from the user while operating the GUI. Each user interaction such as a mouse click is recorded and stored. Recorded data are: user name, timestamp, GUI component (e.g. solution view), and concrete interaction (e.g. mouse click on ‘open most likely solution’ for the symptom ‘Robot has stopped’ at ‘KUKA LBR iiwa 14 R820’).

3.6. Prototype Implementation

The components have been implemented prototypically, serving as a demonstrator for the evaluation. The front end is programmed in Typescript using Angular2+ and Material-Design-Light. The back end uses Java as programming language in combination with the Spring library. To enable fast querying, Apache Solr has been selected as technology for implementing the ontology. The ontology has been filled with demo data. User feedback is stored in a PostgreSQL relational database. Communication between the individual components is realized using the technologies Web socket, RESTFul, and Spring Message Routing. The robot is connected via OPC UA.

4. Evaluation

We evaluate the method and prototypical implementation presented in Section 3 by comparing it with the requirements specified in Section 2.

R1 - *Resolve error quickly and with little effort* - As shown in Section 3.1, a user can navigate from an error alert to an appropriate solution with a single interaction. In contrast to manual lookups without connection to the machine or even paper-based approaches, the proposed methodology can be assumed to enable the personnel to resolve errors more quickly and with reduced effort. Another aspect indicating the fulfilment of this requirement is the reduction of possible user typing errors.

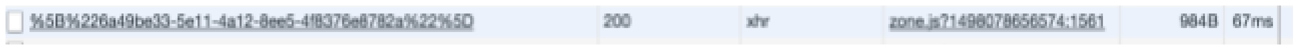
R2 - *Appropriate technical documentation* - The appropriateness of the solution presented is dependent on the users' preference, including his skill level, and the availability of appropriate documentation. The solution proposed in Section 3 offers functionalities for collecting user feedback and for storing it in the user feedback store (Section 3.5). After querying the ontology for technical documentation fragments (SCS), the user feedback store is consulted prior to re-ordering the SCSs (Section 3.4). In addition, the ontology can include information regarding the target user group (Section 3.3). Hence, the appropriateness of the documentation relies on both, the ontology and the user feedback store. Using the demo use case from the introduction alongside with demo data, the solution shown in Figure 2 is appropriate for a junior application developers skill level for programming a LBR iiwa R820, hinting towards the fulfilment of R2.
For fully evaluating requirements R1 and R2, a comprehensive survey with factory personnel and real technical documentation needs to be conducted. This is planned as future work.

R3 - *Match the machine context* - The ontology is consulted as soon as a connected machine has communicated its error event and context. As stated in Section 3.3, the ontology acts as bridging mechanism between the robot and the technical documentation. Hence, a properly filled ontology is a prerequisite for mapping the machine context to technical documentation. The ontology contains relationships between products, errors and modularized technical documentation (SCS) and is used for querying SCSs matching both the error and machine context. Thus, it can be assumed that R3 is met.

R4 - *Automatically* - As stated in Section 3.1 and shown in Figure 1, the proposed methodology includes a navigation component that provides an automatic shortcut towards technical documentation with the most likely solutions. In favour of providing an overview of all connected machines and errors occurring on them, the concept includes a dashboard view where the navigation component is provided per machine error. Hence, the proposed solution includes an automatic path towards error solutions and fulfils R4.

R5 - *Devices* - The layered software architecture provides application programmers interface (API) between presentation and logic layer. GUIs for different device classes can be implemented, thus providing the basis for fulfilling R5. As explained in Section 3.6, the demonstrator is programmed using the Material-Design-Light library. Its grid mechanism is used while implementing the prototype, resulting in a responsive web page. It can be used in a web browser, on a tablet PC, and on a smartphone.

R6 - *Fast interaction* - The methodology presented in Section 3 can be implemented efficiently, allowing response times below 1 s for user interactions. Employing the prototype implementation (Section 3.6), an initial performance measurement has been conducted. A screenshot of the web browsers' debug console is shown in Figure 5, displaying an end to end response time of 67 ms for a symptom lookup. The test was conducted using the prototype locally, running on a MacBook Pro (Late 2016, i7-6567U @ 3.30GHz, 16 GB RAM) on battery, with a limited amount of test data.



```
%5B%226a49be33-5e11-4a12-8ee5-4f8376e8782a%22%5D    200    xhr    zone.js?1498078656574:1561    984B  67ms
```

Figure 5: Symptom lookup in 67 ms; using the prototype locally.

5. Related Work

Bunte et al. (2016) introduced an interface powered by natural language with the purpose of interacting with a smart device (system). Their solution is suitable for answering questions like "Are there any anomalies in the system?" by combining different techniques including natural language processing, ontologies, analytic and synthetic algorithms organized hierarchically. However, they do not provide means for providing an overview of machine status as the dashboard component introduced in Section 3 does. In addition, their solution relies on the user typing in their question, opening the possibility for misspelling. Our solution offers more automation thanks to the navigation component towards a technical documentation which includes solutions.

Hornung et al. (2014) introduced a new kind of semi-supervised robot anomaly detection, using a data-driven approach. By utilizing cascading machine learning techniques, their solution is able to detect anomalies while little information is present. The employed machine learning techniques are maps of valid and invalid data combined with a distance measurement and a Support Vector Machine. The technique of detecting anomalies from raw data in situations where little information is present could be an addition for the pure event-based approach described in Section 3. In contrast to the authors' proposed method in Section 3, their approach does not include a mapping of technical documentation to the detected errors/anomalies.

The CSC project (CyberSystemConnector, 2017) introduces an approach for storing technical documentation on individual machines or even machine parts in a smart factory. In contrast, we propose a central storage for technical documentation. We argue that the centralized approach eases the editing and updating process of technical documentation to a great extent. In case the information is stored in a distributed manner, i.e. each machine is carrying its own documentation, the need for an individual updating mechanism arises. Wang et al. (2017) compare a client–server paradigm with a mobile agent paradigm in the context of predictive maintenance, outlining pros and cons. Our approach follows the client-server paradigm whereas the CSC project follows the agent paradigm. Wang et al. confirm our view that resource management is eased following the client-server paradigm.

A concept regarding knowledge acquisition and process mapping in context of a smart factory is proposed by Panfilenko et al. (2016). By combining a semantic media wiki with process knowledge in BPMN and techniques of object character recognition (OCR) on hand-written incident reports, their solution is capable of mapping detected anomalies onto existing knowledge. Their demonstrator focuses on manually collected incident reports and does not implement an automatic connection to the machine as the authors' solution does.

In accordance with the proposed software architecture, numerous publications in the context of smart factory propose a layered architecture. Examples are the cyber-physical systems architecture for Industry 4.0-based manufacturing systems (Lee et al., 2015), the reference architecture model for Industry 4.0 (DIN, 2016), and the event-driven manufacturing information system architecture for Industry 4.0 (Theorin et al., 2016).

6. Conclusions and Future Work

When machine errors occur in factories, it is important to act quickly in an appropriate way. In this paper, we have presented a methodology for semantically matching symptoms and causes in error situations, and automatically presenting solutions to end-users in an intuitive way. The methodology has been implemented prototypically. As stated in the evaluation section 4, the first prototype shows promising results.

To implement the proposed concept for a more production like environment, a number of next steps are required. The ontology needs to be filled with comprehensive technical documentation. An interface to a state-of-the-art knowledge management system is currently being implemented, providing the ontology. A thoroughly conducted study is planned which will compare the stand-alone knowledge management system with our integrated, context-aware solution. Learnings from the study will be used to improve the solution before it can replace an existing stand-alone system in production.

A future addition could be the incorporation of automatic fixes for common machine errors. Users could then be offered an additional option: 'automatically apply solution'. May our contribution help improve the productivity and efficiency of smart factories.

References

An, K. N., Kaufman, K. R. and Chao, E. (1989). "Physiological considerations of muscle force through the elbow joint", In: *Journal of Biomechanics*, vol. 22, 11-12, pp. 1249-1256. doi: 10.1016/0021-9290(89)90227-3

Bauernhansl, T., Hompel, M. ten and Vogel-Heuser, B., eds. (2014). *Industrie 4.0 in Produktion, Automatisierung und Logistik: Anwendung, Technologien, Migration*, Wiesbaden, Springer Vieweg. doi: 10.1007/978-3-658-04682-8

Bitkom (2017). *IT-Unternehmen bauen Angebote für die Industrie 4.0 aus* [Online]. Available at https://www.bitkom.org/Presse/Presseinformation/IT-Unternehmen-bauen-Angebote-fuer-die-Industrie-40-aus.html (Accessed 21 March 2017).

Bunte, A., Diedrich, A. and Niggemann, O. (2016). "Integrating semantics for diagnosis of manufacturing systems", In *IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA).* Berlin, Germany, 06/09/2016 - 09/09/2016, IEEE, pp. 1-8. doi: 10.1109/ETFA.2016.7733721

Busse, J., Humm, B., Lubbert, C., Moelter, F., Reibold, A., Rewald, M., Schluter, V., Seiler, B., Tegtmeier, E. and Zeh, T. (2015). "Actually, What Does "Ontology" Mean?: A Term Coined by Philosophy in the Light of Different Scientific Disciplines", In: *Journal of Computing and Information Technology*, vol. 23, no. 1, p. 29. doi: 10.2498/cit.1002508

CyberSystemConnector (2017). Maschinendokumentation intelligent erstellen und nutzen [Online]. Available at https://www.cyber-sc.de/site/ (Accessed 8 June 2017).

DIN (2016) *SPEC 91345:2016-04: Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)*, Berlin: Beuth.

Garrett, J. J. (2011). *The elements of user experience: User-centered design for the Web*, 2nd ed, Berkeley, Calif., New Riders.

German Federal Ministry of Education and Research (2017). Industrie 4.0 Standortverteilung Deutschland [Online]. With filter active projects. Available at https://www.bmbf.de/de/zukunftsprojekt-industrie-4-0-848.html (Accessed 5 May 2017)

Henninger, K., Wahlster, W. and Helbig, J. (2013). *Recommendations for Implementing the Strategic Initiative Industrie 4.0: Final Report of the Industrie 4.0 Working Group* [Online]. Available at http://www.acatech.de/fileadmin/user_upload/Baumstruktur_nach_Website/Acatech/root/de/Material_fuer_Sonderseiten/Industrie_4.0/Final_report__Industrie_4.0_accessible.pdf (Accessed 13 January 2017).

Hornung, R., Urbanek, H., Klodmann, J., Osendorfer, C. and van der Smagt, P. (2014). "Model-free robot anomaly detection", *IEEE/RSJ International Conference on Intelligent Robots and Systems 2014*, pp. 3676-3683.doi: 10.1109/IROS.2014.6943078

Industrial Internet Consortium (2014). *About Us* [Online]. Available at http://www.iiconsortium.org/about-us.htm (Accessed 8 June 2017).

French Embassy in Berlin (2015). *Slidset: Industrie du Futur* [Online]. Available at http://www.ambafrance-de.org/Vorstellung-des-neuen-franzosischen-Plans-Industrie-du-Futur-in-der-Botschaft .

ISO (2006). *DIN ISO 9241-110:2006: DIN EN ISO 9241-110 Ergonomie der Mensch-System-Interaktion - Teil 110: Grundsätze der Dialoggestaltung (ISO 9241-110:2006); Deutsche Fassung EN ISO 9241-110:2006: Perinorm* [Online]. Available at http://perinorm-s.redi-bw.de/volltexte/CD21DE04/1464024/1464024.pdf (Accessed 28 June 2013).

ISO (2010). *DIN ISO 9241-210:2010: Ergonomie der Mensch-System-Interaktion - Teil 210: Prozess zur Gestaltung gebrauchstauglicher interaktiver Systeme* [Online]. Available at http://perinorm-s.redi-bw.de/volltexte/CD21DE05/1728173/1728173.pdf (Accessed 29 October 2014).

Kaupp, L., Beez, U., Humm, B. G. and Hülsmann, J. (2017). "From Raw Data to Smart Documentation: Introducing a Semantic Fusion Process for Cyber-Physical Systems", In: *Proceedings of the 2017 Collaborative European Research Conference (CERC 2017)*, Karlsruhe, Germany.

Koffka, K. (2014). *Principles of Gestalt psychology*, Mimesis Edizioni.

KUKA (2017). *Product Brochure: Sensitive robotics_LBR iiwa* [Online]. Available at https://www.kuka.com/-/media/kuka-downloads/imported/9cb8e311bfd744b4b0eab25ca883f6d3/kuka_lbr_iiwa_broschuere_de.pdf (Accessed 23 May 2017).

Lee, J., Bagheri, B. and Kao, H.-A. (2015). "A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems", In: *Manufacturing Letters*, vol. 3, pp. 18-23. DOI: 10.1016/j.mfglet.2014.12.001 (Accessed 19 May 2017).

Lettnin, D. and Winterholer, M. (2017). *Embedded software verification and debugging*, New York, NY, Springer. doi: 10.1007/978-1-4614-2266-2

Lin, L. and Lee, H.-M. (2012). "Machine Failure Diagnosis Model Applied with a Fuzzy Inference Approach", in *Watada, J. (ed) Intelligent decision technologies: Proceedings of the 3rd International Conference on Intelligent Decision Technologies (IDT2011)*, Berlin, New York, Springer, pp. 185-190. doi: 10.1007/978-3-642-22194-1_19

Nguyen, V.-A.-Q. (2017). "Study on realtime control system in IoT based smart factory: Interference awareness, architectural elements, and its application", In: *2017 Seventh International Conference on Information Science and Technology (ICIST).* Da Nang, Vietnam, IEEE, pp. 25-28.doi: 10.1109/ICIST.2017.7926774

Panfilenko, D., Poller, P., Sonntag, D., Zillner, S. and Schneider, M. (2016). "BPMN for knowledge acquisition and anomaly handling in CPS for smart factories", In: *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA).* Berlin, Germany, 06/09/2016 - 09/09/2016, IEEE, pp. 1-4.doi: 10.1109/ETFA.2016.7733686

Saalmann, P. and Hellingrath, B. (2016). "Coordination in Spare Parts Supply Chains", In: *2016 International Conference on Collaboration Technologies and Systems (CTS).* Orlando, FL, USA, IEEE, pp. 601-604.doi: 10.1109/CTS.2016.0111

Starke, G. (2015). *Effektive Software-Architekturen: Ein praktischer Leitfaden*, 7th edn, München, Hanser. doi: 10.3139/9783446444065

Theorin, A., Bengtsson, K., Provost, J., Lieder, M., Johnsson, C., Lundholm, T. and Lennartson, B. (2016). "An event-driven manufacturing information system architecture for Industry 4.0", In: *International Journal of Production Research*, vol. 55, no. 5, pp. 1297-1311. DOI: 10.1080/00207543.2016.1201604 (Accessed 22 May 2017).

W3C (2014) *RDF 1.1 Concepts and Abstract Syntax* [Online]. Available at https://www.w3.org/TR/rdf11-concepts/ (Accessed 16 June 2017).

Wang, J., Zhang, L., Duan, L. and Gao, R. X. (2017). "A new paradigm of cloud-based predictive maintenance for intelligent manufacturing", In: *Journal of Intelligent Manufacturing*, vol. 28, no. 5, pp. 1125-1137. DOI: 10.1007/s10845-015-1066-0.

Zhang, C. and Ordóñez, R. (2012). *Extremum-seeking control and applications: A numerical optimization-based approach*, New York, London, Springer. doi: 10.1007/978-1-4471-2224-1