Recommendation function for smart data analytics toolbox to support semantic merging of middle-of-life data streams

Eddahab, Fatima Zahra Abou; Horvath, Imre

DOI
10.1109/ICCAD49821.2020.9260510

Publication date
2020

Document Version
Final published version

Published in
2020 International Conference on Control, Automation and Diagnosis, ICCAD 2020 - Proceedings

Recommendation function for smart data analytics toolbox to support semantic merging of middle-of-life data streams

Fatima-Zahra Abou Eddahab, Imre Horváth
Department of Sustainable Design Engineering
Delft University of Technology
Delft, The Netherlands
{f.aboueddahab, i.horvath}@tudelft.nl

Abstract*—Continuous enhancements of connected products make them able to generate and communicate a huge amounts of middle-of-life data streams to their producers. This affordance also creates a challenge for current data analytics tools unable to keep up with the heterogeneous nature and characteristics of these type of data. Accordingly, a function able to combine data from multiple data streams and analyze them as one source of information is definitely needed in a next-generation data analytics toolbox to support product enhancements by designers. As a result of a recent Ph.D. project, this paper presents the conceptualization and the implementation of a novel function of merging middle-of-life data streams. The implemented computational mechanism (i) acquires middle-of-life data streams, (ii) pre-processes them individually, (iii) merges information from the concerned streams, (iv) derives recommendation based on the merged information, and (v) send a recommendation as a message to the designer. The performance of the computational implementation was tested in an application case of data steaming and management to white goods designers for enhancing a connected washing machine. From a computational point of view, the testing proved that the set of proprietary algorithms designed for the realization of computational merging, together with the existing ones taken from the literature, were able to efficiently perform the subtasks. The advantages of merges were: (i) it provides more information than the one obtained by processing sensors' data individually, (ii) it reflects the condition of the product with a higher fidelity, (iii) it communicates information about the product while it is in use by the customer, (iv) it reduces the sensors analyses time and effort, and (v) it provides recommendation as an action plan concerning the product at hand. The outcomes of this study will be used in a follow up research to develop a comprehensive smart data analytics toolbox to support product designers in product innovation.

Keywords—data analytics; middle-of-life data; data merging; semantic interpretation; product designers; white goods

I. INTRODUCTION

During the last decades, data merging has become a rapidly evolving topic in various application fields [1]. It is defined as a synthesis of information provided by multiple data sources. Its objective is to establish a relatively consistent and complete description through a more complete and accurate set of information [2]. The need for data merging, especially for semantic fusion of middle-of-life data streams (MoLD-Ss) was reported by product designers in an investigation that we conducted to determine the needs, satisfaction and expectations of white goods designers concerning a next-generation smart data analytics toolbox (SDATB) [3]. Designers concretely required (i) semantic interpretation of data analytics outputs, as well as (ii) merging different data streams from different sources. To explore the need in a multi-disciplinary manner, we combined five implicative theories based on the principles of the axiomatic theory fusion (ATF) approach [4]. The specific theories considered for fusing were dedicated to (i) professional needs of product designers, (ii) advanced technological enablers, (iii) evolution of data analytics, (iv) combined creative problem solving and decision-making, and (v) functional and structural interoperability of enablers. The process of the ATF consisted of five main stages, (i) selecting theories based on their usefulness as source theories, (ii) axiomatic discretization of component theories which consists of semantic discretization of theories, and arrangement and composition of axioms and postulates structures, (iii) semantic and visual capturing of relationships that is done in three steps: creation of relationship network, matrix representation and rearrangement, and deriving propositions in a given context, (iv) actual fusion of the component theories that is done in three steps: syntactic processing and merging of component theories, deriving propositions based on units of resultant theory, and transferring propositions into a narrative description, finally (v) validation of the new theory in the context of the planned application. In this sense The results of the ATF confirmed the relevance of the hypothesis that semantic merging of MoLD-Ss and offering recommendation to designers based on combined data streams is a computational function that needs to be provided a SDATB.

According to published works, implementation of a multi-sensor-based data merging approach (i) improves the probability of proper detection, (ii) extends the spatial and temporal coverage, (iii) reduces ambiguity, (iv) enhance systems reliability, and (v) increases system robustness [5]. However, computational is a complicated task, especially when semantic merging is targeted. This is a so-called high-level merging and is difficult to realize for two reasons [6]. First, inferring semantic knowledge needs the transformation of a low-level data/information into higher-level ones, which typically suffers from information deficit. Second, understanding semantics by a system requires the ability of (i) working towards a set purpose,

978-1-7281-6999-6/20/$31.00 ©2020 IEEE

(i) acquiring context awareness, and sharing pertinent knowledge. These characteristics are typically thought of as features of human beings.

The proposed merging function complements semantic merging by offering an action plan to the designer. This will be evidenced in the rest of the paper. Section 2 presents the recommendation function supporting the realization of semantic merging. Section 3 focuses on the conceptualization from a computational perspective, including both a functional specification and a conceptual architecture. Section 4 deals with the specification and implementation of the algorithms. Section 5 demonstrates the use of the developed function in an application case. It includes the setting up, the conduct, the results of the testing and interpretation of the computational performance. Finally, Section 6 discusses the outcomes of the reported work, concludes about the overall findings, and presents the plans for short-term and longer term research activities.

II. RECOMMENDATION FUNCTION FOR MERGING MIDDLE-OF-LIFE DATA STREAMS

The computational function that we have developed not only computationally fuses MoLD-Ss but also provides recommendations to the designer about what to do and how to do with the merged data streams. Therefore, it is named “recommendation for merging middle-of-life data streams”., Below, we refer to it by the symbol F_{SB1}, where the subscript SB stands for ‘smart basic’ function. The basic functions are operational functions that are directly related to data analytics. As such, they are derived from fundamental requirements, which need to be satisfied by the SDATB. They do not include data management, interaction, or communication operations. In the above symbol, index “1” indicates that it is the first one of the many computational functions implemented for the SDATB.

Computationally, F_{SB1} is supposed to merge semantically interrelated MoLD-Ss generated by different sensors of the same product. By doing so, it facilitates gaining additional information and deriving knowledge from the data streams to support decision-making in various contexts of product enhancement. The distinct data streams may complement each other. Thus, the merged MoLD-Ss can provide additional semantic information that is initially not conveyed by any one of the MoLD-Ss. However, eliciting semantic information requires appropriate inferring techniques. The proposed function adopts the principles of high-level multi-sensor data merging. It contextualizes the information conveyed by MoLD-Ss and analyzes their meanings in that context. Eventually, the recommendations concerning possible enhancement opportunities are based on the pieces of information generated by context-based reasoning. This content of recommendation is deduced by analyzing the outcomes of the merging in the specific context and is displayed for the designer as a displayed message.

To realize the proposed function, a neural network-based approach has been adapted to merging MoLD from multiple different sources. The fact of the matter is that there are specific implementations of neural networks (NNs) that are appropriate to this purpose [7]. In addition, also devising a recommendation can be done by using a NN-based approach [8]. In order to find the neural network that can predict the required output dependably, an optimization process has been proposed considering gradient descent [9]. This entails the use of a multi-layered NN that can be efficiently trained with stochastic gradient descent. Given a fixed amount of data, generalization can be improved by changing the architecture of the neural network.

III. CONCEPTUALIZATION OF THE COMPUTATIONAL MERGING OF MIDDLE-OF-LIFE DATA STREAMS

The realization of F_{SB1} required the specification of three elements: (i) the inputs provided by the designers (MoLD-Ss), (ii) the outputs expected from the function, and (iii) the computational procedures to execute $F_{SB1.}$ The expected outputs were the message(s) displayed to designer about the resultant MoLD-S and the action plan. The first step of realization was the decomposition of the function into a lower level functions and elementary functions considering a possible computational workflow. The decomposition resulted in five sub-functions: (i) acquiring real-time sensor MoLD-Ss ($F_{SB1,1}$), (ii) individually pre-processing the selected MoLD-Ss ($F_{SB1,2}$), (iii) merging information from the chosen middle-of-life data streams ($F_{SB1,3}$), (iv) deriving recommendation based on the captured information (anomalies) ($F_{SB1,4}$), and (v) formulating the recommendation as a message for the designer ($F_{SB1,5}$).

The sub-function $F_{SB1,1}$ locates the product sensors and forwards their MoLD-Ss to the SDATB. The forwarded data streams may be stored on the background storage devices of the computer hosting the SDATB, or on a separate storage device (in the cloud). In order to get a confirmation from the designer, the sub-function $F_{SB1,2}$ visualizes the data streams for the designer using various means offered by the SDATB (for example, plots and histograms). In addition, it pre-processes the single-modality data streams by selecting and applying particular processing rules. In the case of complicated data streams with unknown patterns, a comprehensive structural pre-processing (filtering or ordering) is applied. In the case of less complicated data streams, the sub-function $F_{SB1,2}$ reduces to data normalization. The computational merging of the MoLD-Ss is eventually done by the sub-function $F_{SB1,3}$. The principle of fusion is correlating the data in the streams based on their time stamps. First, the sub-function generates intermediate representations to reduce time-dependent data into a compact fixed length vector. Then, it combines the data streams and generates a behavior descriptor according to the merged MoLD-Ss.

With a view to facilitating the application of machine learning, the sub-function $F_{SB1,3}$ embeds the fused MoLD-Ss in a latent space (also called hidden space) of the neural network that is used for machine learning. In this space, data are mapped so as the similar data points are closer to each other. The latent space representations can be used to transform complex forms of raw data into simpler forms that are easier to analyze. The mapping to the latent space also help cluster similar cases. The sub-function $F_{SB1,4}$ (i) detects anomalies in the merged data streams, (ii) matches the anomalies to the pre-programed knowledge in the SDATB, (iii) orders them according to their similarity, (iv) makes a report on each of the ordered anomalies based on the merged MoLD-Ss, and (v) converts the outcome into a specific recommendation. Sub-function $F_{SB1,5}$ (i) retrieves a template for message construction, (ii) constructs a recommendation message for the designer accordingly, (iii) uses the retrieved template to construct the message, and (iv) communicates the message to the designer. In line with the merged data, the message informs the designers about what is improper concerning the product. As a complement of the

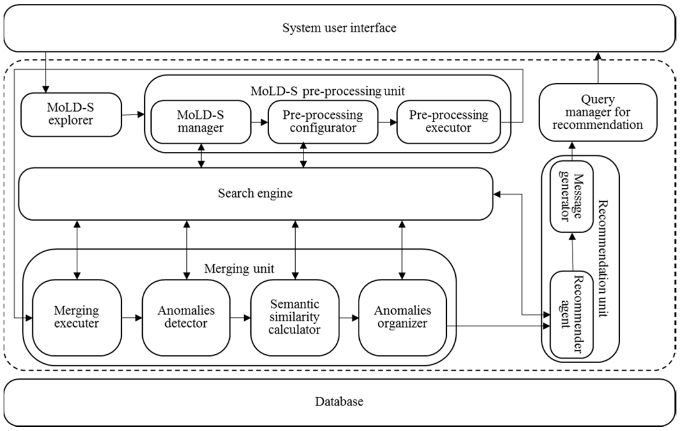

Fig. 1. The overall architecture of the recommendation function for merging middle-of-life data streams

recommendation, a plan of actions is generated to resolve the detected operational anomalies of the product.

As a next step towards the computational algorithms of F_{SB1}, a conceptual architecture was specified. The overall architecture of F_{SB1} is shown in Fig. 1. The main constituents are: (i) the user interface, (ii) the database, (iii) the anomaly explorer, (iv) the pre-processing unit (including the manager, the configurator and the executor), (v) the merging unit (including the executor, the detector, the similarity calculator, and the organizer), (vi) the query manager, (vii) the search engine, and (viii) the recommendation unit (including the generator and the agent). The user interface enables the communication between the designer and the SDATB and transfers the inputs and the outputs to and from the toolbox. The database (also referred to as knowledge warehouse) stores (i) the data streams, (ii) the rules and the conditions for analyses, and (iii) the results of merging data streams. The MoLD-Ss explorer is responsible for the exploration of the data streams to be analyzed. The MoLD-Ss pre-processing unit communicates with the designer, and receives and processes the individual MoLD-Ss. The MoLD-Ss manager visualizes the data streams stored in the database and makes them available for the search engine. The pre-processing configurator determines the pre-processing rules and conditions to be applied to the individual streams by the pre-processing executor. These two components use the knowledge stored in the database. The pre-processed MoLD-Ss are transferred to the merging unit. Its semantic similarity calculator compares the explored anomalies with those stored to determine resemblances. The anomalies organizer manages the weights and, based on them, filters and organizes the anomalies for the recommendation generator. The recommender agent converts the information generated by the mentioned components into recommendation contents. The message generator produces messages to the designer using the recommendation contents. Finally, the query manager converts the generated recommendation message to human language and communicates it in this form to the designer.

IV. Implementation of the Algorithms for Merging Middle-of-Life Data Streams

The computational workflow (CWF), which shows the operational relationships among the algorithms, is shown in Fig. 2. The computational implementation of merging of MoLD-Ss needed twenty algorithms of various complexities. Three of them were needed to realize the sub-function $F_{SB1,1}$. Algorithm A_1 of the SDATB requests the list of sensors to be analyzed by

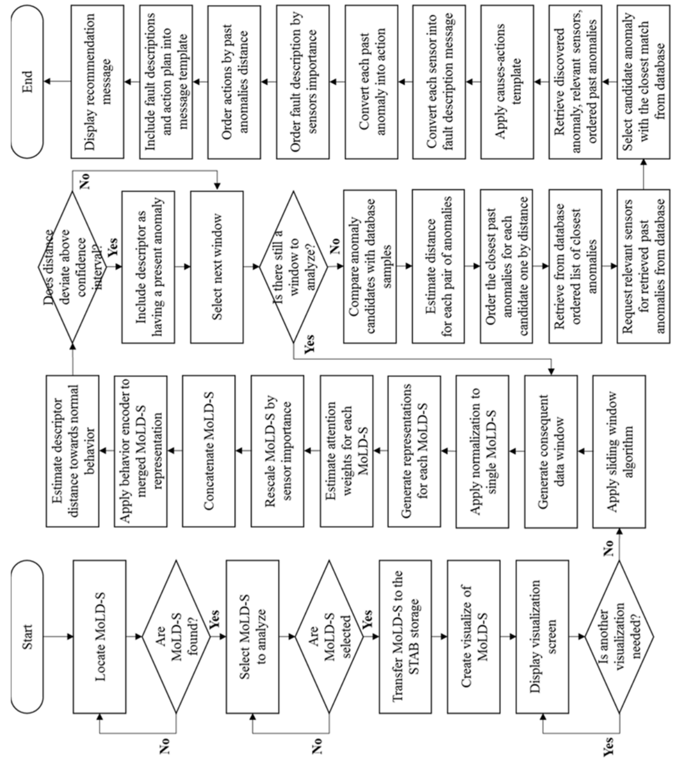

Fig. 2. The computational workflow of the merging of middle-of-life data streams

from the designer. Algorithm A_2 accesses the location and the sources of data streams. Algorithm A_3 acquires the MoLD-Ss either from a remote storage (for example, a cloud environment) of from the local storage the SDATB. For the sub-function $F_{SB1,2}$, two algorithms were needed. Algorithm A_4 provides visualization means (plotting) to help comprehend the data despite their raw format. Algorithm A_5 normalizes the MoLD-Ss to make a proper use of the data streams in further analyses possible. It removes the anomalies such as deleting data (e.g. removing correlated time series) that might complicate the analysis. This algorithm (i) inserts additional information (e.g. by applying one hot encoding for categorical features), or (ii) updates existing data (e.g. clipping outliers).

For the sub-function $F_{SB1,3}$, four algorithms were needed. Algorithm A_6 processes the normalized MoLD-Ss time series using a statistical model. This is needed to generate a length invariant representation of MoLD-Ss and, this way, to reduce the computational overheads in the follow up steps. In order to provide a recommendation based on a multi-stream dataset, we annotate past anomalies with descriptions called labels. The unique labels assigned to the anomalies are used for clustering a predefined set of classes. For the implementation of the computational function, the triplet loss function was considered for the neural network, Furthermore, for each anomaly, we defined a set of incidents to provide sufficient data for training the model but to avoid overfitting. The triplet loss training is capable to fit a dataset of 8 million unique labels and to achieve higher than 95% of classification accuracy [10]. A dataset has been used to train the neural network architecture. We used a stochastic gradient descent training with mini-batches and the loss function is a triplet loss. Our model implementation assumes that it operates in a sliding window fashion along time axis to encode behavior patterns and find the closest match among past cases.

The neural network chosen for the purpose of computations realizes Algorithm A_7, that is used to predict the importance weights of the sensors for forward pass (which influence the calculation process as well as the values of the output layers generated based on the input data). Algorithm A_7 calculates or estimates the importance of the sensors from the perspective of the analyzes. It is needed only in those cases in which the data of a large number of sensors are to be merged and analyzed. To build this algorithm we needed to define the following: (i) X is a real valued matrix of BxMxT size, (ii) M represents multi-modal features of each window of frames (sliding window), (iii) T represents the time frame, and (iv) B is the batch size. Algorithm A_7 is presented below. By considering a lower number of relevant data streams, the interpretative predictions are improved. The outcomes of this steps are used by Algorithm A_8, which merges the data streams (obtained from various sensors and captured in the same time frame). Algorithm A_8 also considers the weights allocated to the sensors and selects for merging those ones, which have the highest weight values. In other words, it orders the sensors according to the estimated fusion weights and considers only a the most relevant portion of MoLD-Ss in the merging procedure. Algorithm A_9 jointly processes the MoLD-S and embeds information in a new latent space, in which a distance reflects a measure of semantic similarity. The behavior descriptor is sensor and source independent.

Algorithm 7. Estimate the importance of sensors

Inputs: I1 = Matrix X

I2 = B

Outputs: O1= h, latent representation of behavior described by the current window of features

O2= a, importance of sensor

```
1: t2 ← conv1(X);
2: t16 ← leaky_relu(t2,0.2);
3: t15 ← conv2(t16);
4: t7 ← leaky_relu(t15,0.2);
5: t6 ← attention(t7);
6: t8 ← reshape(t6,B,M,1).* t7;
7: t9 ← behavior_conv1(t8);
8: t10 ← tanh(reshape(sum(t9,2),B,1,L));
9: Return struct('h', t10, 'a', t6)
```

For the computational implementation of sub-function $F_{SB1,4}$, six algorithms were needed. Algorithm A_{10} estimates the probability of anomalies. It is a preliminary step to a more thorough search through the knowledge database, which contains a list of pre-recorded anomalies. The algorithm calculates the distance to known anomalies in the database. Algorithm A_{11} gathers similar past anomalies from database. It performs a search for similar descriptors by iterating through the pairs of detected anomalies and past anomalies. Algorithm A_{12} calculates the pairwise distance between the detected and the past anomalies. These anomalies are ranked by Algorithm A_{13} and retrieved by Algorithm A_{14}. Algorithm A_{15} executes the semantic merging of the retrieved anomalies. In addition, it generates a recommendation and an action plan concerning the product. Finally, sub-function $F_{SB1,5}$ needs five algorithms to be realized. Algorithm A_{16} selects the template for the recommendation message from the database. Algorithms A_{17} and A_{18} successively convert the detected anomalies and the action plan into the components of the recommendation message. Algorithm A_{19} orders the appearances of the individual anomalies and includes the action plan in the recommendation message. Algorithm A_{20} integrates the ordered components of the message into the template and provides the recommendation message.

TABLE II. EXAMPLES OF MAPPING BETWEEN ANOMALIES, SENSORS AND RECOMMENDATION MESSAGE TABLE STYLES

Anomaly number	Description	Related sensors	Recommended action
Anomaly 1	Mechanical worn out of mostly used components in the washing machine (washing drum, brakes to stop the drum, and related components).	S_1 or S_2 or S_3 or S_4	Mechanical control, adjustment or replacement of components are needed
…	…	…	…
Anomaly 5	Abnormal temperature values, as well as heating time deviation, and potentially sporadic device terminations. This can be caused by overheating, or under-heating issues.	S_{11}, S_{12}, S_{13}	Water heater element should be cleaned or replaced

Due to the serious page limitation in this paper, the above introduced algorithms cannot be detailed further on pseudo-code or instruction level. Notwithstanding this, their use in a particular application case is demonstrated in the next section. According to the operational scenario circumscribed above, after locating the sensors and selecting the data streams for analysis and merging, the data contents of the MoLD-Ss are transferred to the SDATB. It completes the analysis and the merging. As the next step, the data contents are visualized and presented to the designer, who can repeatedly visualize the data streams in various plotted forms. To prepare for the merging, Algorithm A_6 (the sliding window algorithm) is used to iterate over the MoLD-Ss. This algorithm selects a consequent time frame for the data and normalizes the data along the time axis. Afterward, the single-stream encoder part of the used NN is applied, and single-sensor latent representation is generated in the attention layer of the NN. In the next step of the data processing, the single-sensor representation is rescaled according to the importance weights. These rescaled representations are concatenated into a two-dimensional matrix, and the behavior encoder part of the NN is

TABLE I. NORMAL AND FAULTY BEHAVIORS FOR SOME OF THE CONSIDERED SENSORS

Sensor code	Normal behavior	Faulty behavior
S_1	Constant force during the whole washing cycle.	Abnormal force at some moments during the washing cycle.
S_2	Constant force during the whole washing cycle, with a bigger deviation than washing drum force gauge axle.	Abnormal force at some moments during the washing cycle, correlated with S_1 faulty activity.
…	…	…
S_{13}	Same as in S_{11}, a positive voltage for a start of heating state change, and a negative voltage for the end of it.	More changes in heating states.

applied. Furthermore, to find the past anomalies that are at the closest distances to the current descriptor, the toolbox makes a database query. If the distance to past anomalies stored in the database is small, then a confidence interval including the current time window and its descriptor allow selecting the anomaly candidate. Otherwise it is skipped. If the algorithm finds no more window to analyze, then it starts a similarity search. In this context, the descriptors are compared to the ones stored in the database of the SDATB. The distances between the

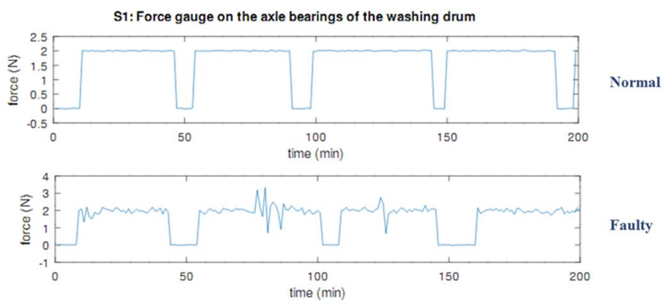

Fig. 3. Visualization of the data streams of the normal and faulty behavior of sensor 1

anomaly pairs are estimated, and the matches are sorted according to the computed distances. Afterward, a ranked list of the anomaly candidates is retrieved based on the contents of the database. In combination with this, the relevant sensors of past anomalies are identified based on the causality matrix. Concerning the anomaly candidates, the one that has the smallest distance to a relevant past anomaly is selected. Concerning the best candidate, this module provides a ranked list of past anomalies and an ordered list of the sensors related to the past anomalies. The latter list is compiled based on the importance weights assigned to the sensors. As a next step, considering the faulty sensors and the possible improvement patterns, this module selects a template for generating a recommendation message. Then, the fault descriptions for each selected sensor and the improvement (or maintenance) actions for each anomaly are retrieved. In a next step, they are arranged according to the importance of the sensors and the anomaly distance values, and used to generate the actual recommendation message, addressing both the contained faults and the action plan. As the very last computational action, this message is displayed to the designer.

V. Validation of the Performance of the Algorithms for Merging Middle-of-Life Data Streams

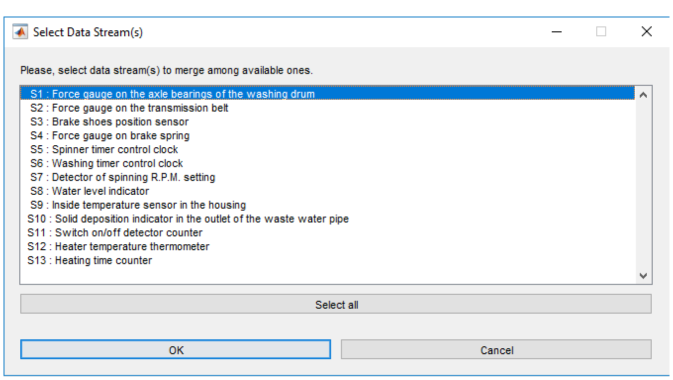

Fig. 4. "Select Data Stream(s)" screen of the merging of middle-of-life data streams module

Functionality validation of the algorithms of the merging middle-of-life data streams module started with the definition of the application context. The chosen reference application case was enhancing a connected washing machine by white goods designer. We assumed that this washing machine had 13 sensors: (i) force gauge on the axle bearings of the washing drum (S_1), (ii) force gauge on transmission belt (S_2), (iii) brake shoes position sensor (S_3), (iv) force gauge on brake spring (S_4), (v) spinner time control clock (S_5), (vi) washing timer control clock (S_6), (vii) detector of spinning R.P.M setting (S_7), (viii) water level indicator (S_8), (ix) inside temperature sensor in the housing (S_9), (x) solid deposition indicator in the outlet of the waste water pipe (S_{10}), (xi) switch on/off detector counter (S_{11}), (xii) heater temperature thermometer (S_{12}), and (xiii) heating time counter (S_{13}). Since we do not have access to real-life data streams, we created artificial data streams (some of them with anomalies, while others without). In addition, we incorporated some prior knowledge about product anomalies. In total, five different sensor failures were considered. To support design changes and enhancement, various action recommendations were generated based on merging multiple MoLD-Ss. Actually, the pieces of information about the anomalies and the related sensors were mapped to recommendation messages. Examples of this mapping are presented in Table I. For the purpose of testing we also generated a more complex but consistent MoLD-Ss. Assumed to be conveyed by the data streams, examples of normal and faulty behaviors are presented in Table II. The characteristic differences between the normal and the faulty behaviors of sensors made the interpretation of MoLD-Ss easier. An example of a visual representations of normal and faulty behavior of sensor 1 is given in Fig. 3.

The algorithms of the merging middle-of-life data streams module were implemented using the resources offered by Matlab. The computationally crucial algorithms were tested in the above-defined application case. To facilitate their testing, we also developed a simple GUI to visualize the outcomes of the algorithms for the designer. The main screen of the GUI includes two actions (two possible buttons to press by the designer) (i) "Data" containing one option called "Select Sensors" to choose the sensors to analyze, since our sensors are already located in the platform, and (ii) "About" which displays general

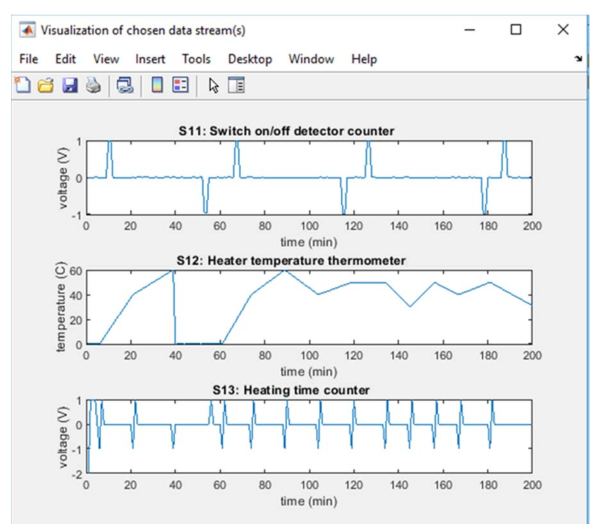

Fig. 5. Combined visualization of the data streams of the sensors 11, 12 and 13

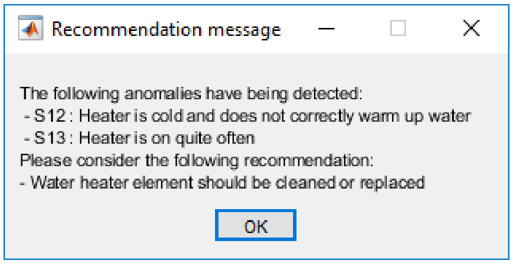

Fig. 6. Recommendation message communicated to the designer

information about the function. Once the designer clicks on the "select sensors" items, he is taken to the next screen, which shows each available MoLD-Ss together with their corresponding codes and a short description.

At this stage of using the SDATB, the designer chooses the sensors he intends to merge (the option "select all" is also available) or to choose only one sensor, if he wants to analyze a particular one. Then, (as shown in Fig. 4), he either presses "OK" to continue with the visualization, or clicks on "Cancel" to go back to the initial screen. After the designer makes his choice, the selected MoLD-Ss are activated (transferred) for analysis. (For the sake of this description, let us assume that the designer selects the sensors S11, S12 and S13). The following screen, called "Visualization", allows him to select the data streams that he wants to see in graphical form. This can be achieved by pressing the button "Visualize". He can choose to see the streams one by one or all of them together plotted alongside. In the latter case, the visualized data streams share the same time axis as illustrated in Fig. 5. This step may be skipped if the designer does not want to check the contents of the MoLD-Ss and starts the merging directly. Once the inspection of represented plots is finished, the designer can return to the previous window by pressing "X" and can select the button dedicated to the initiation of merging data streams. The details of the merging process are not visualized on the GUI. Within seconds, a textual recommendation is presented to the designer.

The recommendation message (i) contains an explanation of the detected anomalies, (ii) identifies the concerned sensors, and (iii) recommends an action plan. The latter is semantically related to the anomalies explored with regard to the different sensors. Shown in Fig. 6, the communicated message is based on the choices assumed above. It can be seen in Fig. 6 that S11 was not mentioned in the message. This means that no anomalous behavior was detected on this particular sensor. If the user of the washing machine would turn it on and off more often, then the sensor S11 would have recorded it. In this case, different recommendation and measures would be advised. It must be noted that not only the water heater element can be an issue, but the whole electrical system of the washing machine can be faulty. To check the appropriateness of the analysis, we repeated the data merging for the same sensors three times. The results evidenced that the algorithm explored the same anomalies and offered the same recommendation.

VI. CONCLUSIONS

The functionality testing proved that this computational mechanism was correctly implemented. From a computational point of view, the integration of the newly designed algorithms and the ones taken from the literature did not lead to any inconsistences. Based on the results shown in Fig. 6, it was observed that reasoning with and learning from the MoLD-Ss, as semantic operations, played a significant role in the formulation of the recommendation messages delivered to the designer. The message could cover not only the detected anomalies, but could also recommend certain actions for the designer. We could conclude that the conditions concerning the conversion of faulty behaviors of the MoLD-Ss into a concrete action plan for the designer were correctly incorporated in the computational mechanism. The function for merging middle-of-life data streams (i) provides more information than that is conveyed by the sensors' data individually, (ii) reflects the condition of the product more realistically, (iii) communicates information about the product while it is in use by the customer (iv) reduces the time and effort of sensor analyses, and (v) provides recommendation as an action plan for the product at hand. Offering this function to product designers will allow them to continually analyze the behaviors of their products and to come up with enhancement solutions in a short while.

However, based on the analysis of the research activities and the testing of the implemented function some limitations were recognized. The lack of publications concerning a comprehensive understanding of the procedure of semantic inferring in the context of product enhancement made it difficult to select and deploy the best algorithms and techniques. The need to incorporate prior knowledge about product anomalies resulted in an inclination in the implementation towards maintenance type of action plans. The development of algorithms which are able to automatically generate rules and to be aware of the dynamic changes in context and data streams, was supported by preprogrammed means. This reduces the time and efforts needed for building scenarios and training the algorithms. The use of simulated MoLD-Ss made it possible to complete various tests, but could not replicate true application cases in which the presence of unexpected data patterns can be assumed. The last observed limitation concerns the usage of the deep learning toolbox of Matlab for the implementation of the computational function. The fact of the matter is that it made implementation process more time consuming in comparison with other computational solutions such as offered by Python in which pre-defined operations can be adapted or even directly used.

REFERENCES

[1] Y. Ban, and A. Jacob, "Object-based fusion of multitemporal multiangle ENVISAT ASAR and HJ-1B multispectral data for urban land-cover mapping," IEEE Trans. Geosci. Remote Sens., vol. 51, no. 4, pp. 1998-2006, April 2013.

[2] W. Zhang, X. Ji, Y. Yang, J. Chen, Z. Gao, and X. Qiu, "Data fusion method based on improved DS evidence theory," BigComp, IEEE, pp. 760-766, 2018.

[3] F. Abou Eddahab and I. Horváth, "What do designers miss regarding the outputs of data analytics tools in the context of possible product improvements?" TMCE Conference, pp. 1-14, May 2018.

[4] I. Horváth, "Combining unmergeables: A methodological framework for axiomatic fusion of qualitative design theories," ICED, Design Society, pp. 3591-3600, August 2019.

[5] S. Gite, and H. Agrawal, "On context awareness for multisensor data fusion in IoT," IC3T, vol. 3, pp. 85-93, March 2016.

[6] L. Berardinelli, V. Cortellessa, and A. Di Marco, "Performance modeling and analysis of context-aware mobile software systems," FASE, Springer, Berlin Heidelberg, pp. 353-367, March 2010.

[7] A. Noureldin, R. Sharaf, A. Osman, and N. El-Sheimy, "INS/GPS data fusion technique utilizing radial basis functions neural networks," PLANS, IEEE, pp. 280-284, April 2004.

[8] T.K. Paradarami, N.D. Bastian, and J.L Wightman, "A hybrid recommender system using artificial neural networks," Expert Syst. Appl., vol. 83, pp. 300-313, 2017.

[9] S. Ruder, S. "An overview of gradient descent optimization algorithms," arXiv preprint arXiv:1609.04747, 2016.

[10] K.Q. Weinberger, J. Blitzer, and L.K. Saul, "Distance metric learning for large margin nearest neighbor classification," Adv. Neural Inf. Process. Syst., 1473-1480, 2006.