Data-Driven Monitoring of Cyber-Physical Systems Leveraging on Big Data and the Internet-of-Things for Diagnosis and Control

Oliver Niggemann[1,3], **Gautam Biswas**[2], **John S. Kinnebrew**[2], **Hamed Khorasgani**[2], **Sören Volgmann**[1] and **Andreas Bunte**[3]

[1] Fraunhofer Application Center Industrial Automation, Lemgo, Germany
e-mail: {oliver.niggemann, soeren.volgmann}@iosb-ina.fraunhofer.de

[2] Vanderbilt University and Institute for Software Integrated Systems, Nashville, TN, USA
e-mail: {john.s.kinnebrew, hamed.g.khorasgani, gautam.biswas}@vanderbilt.edu

[3] Institute Industrial IT, Lemgo, Germany
e-mail: {andreas.bunte}@hs-owl.de

Abstract

The majority of projects dealing with monitoring and diagnosis of Cyber Physical Systems (CPSs) relies on models created by human experts. But these models are rarely available, are hard to verify and to maintain and are often incomplete. Data-driven approaches are a promising alternative: They leverage on the large amount of data which is collected nowadays in CPSs, this data is then used to learn the necessary models automatically. For this, several challenges have to be tackled, such as real-time data acquisition and storage solutions, data analysis and machine learning algorithms, task specific human-machine-interfaces (HMI) and feedback/control mechanisms. In this paper, we propose a cognitive reference architecture which addresses these challenges. This reference architecture should both ease the reuse of algorithms and support scientific discussions by providing a comparison schema. Use cases from different industries are outlined and support the correctness of the architecture.

1 Motivation

The increasing complexity and the distributed nature of technical systems (e.g. power generation plants, manufacturing processes, aircraft and automobiles) have provided traction for important research agendas, such as Cyber Physical Systems (CPSs) [1; 2], the US initiative on the "Industrial Internet" [3] and its German counterpart "Industrie 4.0" [4]. In these agendas, a major focus is on self-monitoring, self-diagnosis and adaptivity to maintain both operability and safety, while also taking into account humans-in-the-loop for system operation and decision making. Typical goals of such self-diagnosis approaches are the detection and isolation of faults and anomalies, identifying and analyzing the effects of degradation and wear, providing fault-adaptive control, and optimizing energy consumption [5; 6].

So far, the majority of projects and papers for analysis and diagnosis has relied on manually-created diagnosis models of the system's physics and operations [6; 7; 8]: If a drive is used, this drive is modeled, if a reactor is installed, the associated chemical and physical processes are modeled. However, the last 20 years have clearly shown that such models are rarely available for complex CPSs; when they do exist, they are often incomplete and sometimes inaccurate, and it is hard to maintain the effectiveness of these models during a system's life-cycle.

A promising alternative is the use of data-driven approaches, where monitoring and diagnosis knowledge can be learned by observing and analyzing system behavior. Such approaches have only recently become possible: CPSs now collect and communicate large amounts of data (see Big Data [9]) via standardized interfaces, giving rise to what is now called the Internet of Things [10]. This large amount of data can be exploited for the purpose of detecting and analyzing anomalous situations and faults in these large systems: The vision is developing CPSs that can observe their own behavior, recognize unusual situations during operations, inform experts, who can then update operations procedures, and also inform operators, who use this information to modify operations or plan for repair and maintenance.

In this paper, we take on the challenges of proposing a common data-driven framework to support monitoring, anomaly detection, prognosis (degradation modeling), diagnosis, and control. We discuss the challenges for developing such a framework, and then discuss case studies that demonstrate some initial steps toward data-driven CPSs.

2 Challenges

In order to implement data-driven solutions for the monitoring, diagnosis, and control of CPSs, a variety of challenges must be overcome to enable the learning pathways illustrated in Figure 1:

Data Acquisition: All data collected from distributed CPSs, e.g. sensors, actuators, software logs, and business data, must meet real-time requirements, as well as including time synchronization and spatial labeling when relevant. Often sensors and actuators operate at different rates, so data alignment, especially for high-velocity data, becomes an issue. Furthermore, data must be annotated semantically to allow for a later data analysis.

Data Storage, Curation, and Preprocessing: Data will be stored and preprocessed in a distributed way. Environmental factors and the actual system configuration (e.g., for the current product in a production system) must also be stored. Depending on the applications, a relational database format, or increasingly distributed noSQL technologies [11], may

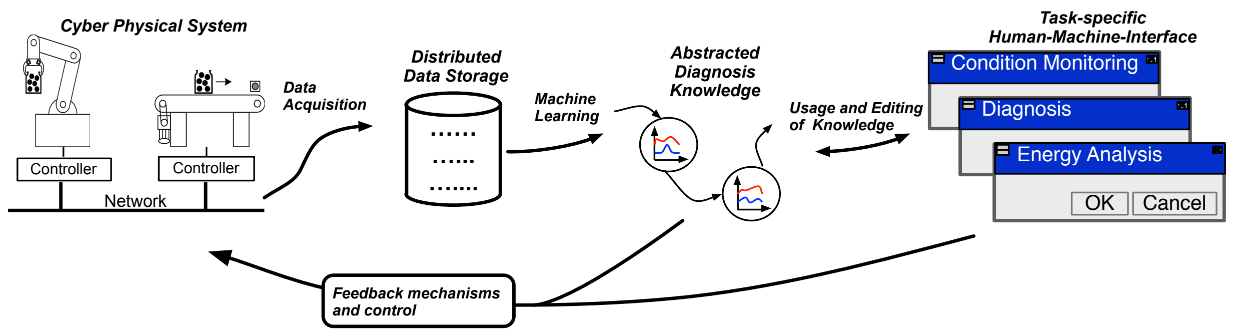

Figure 1: Challenges for the analysis of CPSs.

need to be adopted, so that the right subsets of data may be retrieved for different analyses. Real-world data can also be noisy, partially corrupted, and have missing values. All of these need to be accommodated in the curation, storage, and pre-processing applications.

Data Analysis and Machine Learning: Data must be analyzed to derive patterns and abstract the data into condensed usable knowledge. For example, machine learning algorithms can generate models of normal system behavior in order to detect anomalous patterns in the data [12]. Other algorithms can be employed to identify root causes of observed problems or anomalies. The choice and design of appropriate analyses and algorithms must consider factors like the ability to handle large volumes and sometimes high velocities of heterogeneous data. At a minimum, this generally requires machine learning, data mining, and other analysis algorithms that can be executed in parallel, e.g., using the Spark [13], Hadoop [14], and MapReduce [15] architectures. In some cases, this may be essential to meet real-time analysis requirements.

Task-specific Human-Machine-Interfaces: Tasks such as condition monitoring, energy management, predictive maintenance or diagnosis require specific user interfaces [16]. One set of interfaces may be more tailored for offline analysis to allow experts to interact with the system. For example, experts may employ information from data mining and analytics to derive new knowledge that is beneficial to the future operations of the system. Another set of interfaces would be appropriate for system operators and maintenance personnel. For example, appropriate operator interfaces would be tailored to provide analysis results in interpretable and actionable forms, so that the operators can use them to drive decisions when managing a current mission or task, as well as to determine future maintenance and repair.

Feedback Mechanisms and Control: As a reaction to recognized patterns in the data or to identified problems, the user may initiate actions such as a reconfiguration of the plant or an interruption of the production for the purpose of maintenance. In some cases, the system may react without user interactions; in this case, the user is only informed.

3 Solutions

As Section 4 will show, the challenges from Section 2 reappear in the majority of CPS examples. While details, such as the machine learning algorithms employed or the nature of data and data storage formats can vary, the primary steps are about the same. Most CPS solutions re-implement all of these steps and even employ different solution strategies—raising the overall efforts, preventing any reuse of hardware/software and impeding a comparison between solutions.

To achieve better standardization, efficiency, and repeatability, we suggest a generic cognitive reference architecture for the analysis of CPSs. Please note that this architecture is a pure reference architecture which does not constraint later implementations and introduction of application-specific methods.

Figure 2 shows its main components:

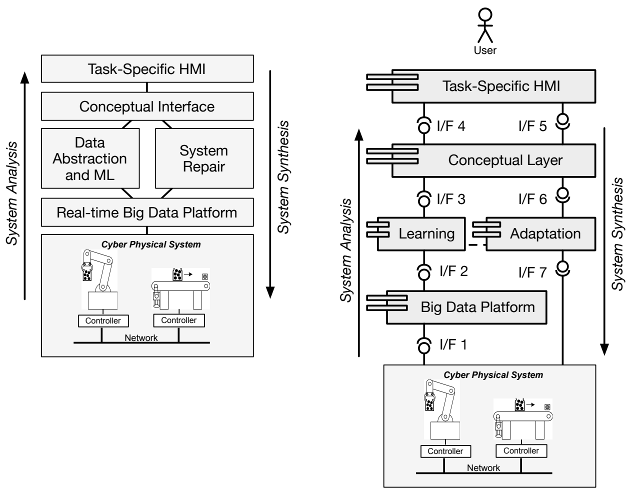

Figure 2: A cognitive architecture as a solution for the analysis of CPSs.

Big Data Platform (I/F 1 & 2): This layer receives all relevant system data, e.g., configuration information as well as raw data from sensors and actuators. This is done by means of domain-dependent, often proprietary interfaces, here called interface 1 (I/F 1). This layer then integrates, often in real-time, all of the data, time-synchronizes them and annotates them with meta-data that will support later analysis and interpretation. For example, sensor meta-data may consist of the sensor type, its position in the system and its precision. This data is provided via I/F 2, which, therefore, must comprise the data itself and also the meta-data (i.e., the semantics). A possible implementation approach for I/F 2 may be the mapping into and use of existing of Big Data platforms, such as Sparks or Hadoop, for storing the data and the Data Distribution Service (DDS) for acquiring the data (and meta-data).

Learning Algorithms (I/F 2 & 3): This layer receives all data via I/F 2. Since I/F 2 also comprises meta-data, the machine learning and diagnosis algorithms need not be implemented specifically for a domain but may adapt themselves to the data provided. In this layer, unusual patterns in the data (used for anomaly detection), degradation effects (used for condition monitoring) and system predictions (used for predictive maintenance) are computed and provided via I/F 3. Given the rapid changes in data analysis needs and capabilities, this layer may be a toolbox of algorithms where new algorithms can be added by means of plug-and-play mechanisms. I/F 3 might again be implemented using DDS.

Conceptual Layer (I/F 3 & 4): The information provided by I/F 3 must be interpreted according to the current task at hand, e.g. computing the health state of the system. Therefore, the provided information about unusual patterns, degradation effects and predictions are combined with domain knowledge to identify faults, their causes and rate them according to the urgency of repair. A semantic notation will be added to the information, e.g. the time for next maintenance or a repair instruction, which will be provided at I/F 4 in a human understandable manner. From a computer science perspective, this layer provides reasoning capabilities on a symbolic or conceptual level and adds a semantic context to the results.

Task-Specific HMI (I/F 4 & 5): The user is in the center of the architecture presented here, and, therefore, requires task-, context- and role-specific Human-Machine-Interfaces (HMIs). This HMI uses I/F 4 to get all needed analysis results and presents them to the user. Adaptive interfaces, rather than always showing the results of the same set of analyses, could allow a wider range of information to be provided, while maintaining efficiency and preventing information overload. Beyond obvious dynamic capabilities like alerts for detected problems or anomalies, the interfaces could further adapt the information displayed to be more relevant to the current user context (e.g. the user's location within a production plant, recognition of tasks the user may be engaged in, observed patterns of the user's previous information-seeking behavior, and knowledge of the user's technical background). If the user decides to influence the system (e.g. shutdown of a subsystem or adaptation of the system behavior), I/F 5 is used to communicate this decision to the conceptual layer. Again, I/F 4 and I/F 5 might be implemented using DDS.

Conceptual Layer (I/F 5 & 6): The user decisions will be received via I/F 5. The conceptual layer will use the knowledge to identify actions which are needed to carry out the users' decisions. For example, a decision to decrease the machine's cycle time by 10 % could lead to actions such as decreasing the robot speed by 10 % and the conveyor speed by 5 % or the decision to shutdown a subsystem. These actions are communicated via I/F 6 to the adaption layer.

Adaption (I/F 6 & 7): This layer receives system adaption commands on the conceptual level via I/F 6—which again might be based on DDS. Examples are the decrease of robot speed by 10 % or a shutdown of a subsystem. The adaption layer takes these commands on the conceptual level and computes, in real-time, the corresponding changes to the control system. For example, a subsystem shutdown might require a specific network signal or a machine's timing is changed by adapting parameters of the control algorithms, again by means of network signals. I/F 7 therefore uses domain-dependent interfaces.

4 Case Studies

We present a set of case studies that cover the manufacturing and process industries, as well as complex CPS systems, such as aircraft.

4.1 Manufacturing Industry

The modeling and learning of discrete timing behavior for manufacturing industry (e.g., automative industry) is a new field of research. Due to the intuitive interpretation, Timed Automata are well-suited to model the timing behavior of these systems. Several algorithms have been introduced to learn such Timed Automata, e.g. RTI+ [17] and BUTLA [18]. Please note that the expert still has to provide structural information about the system (e.g. asynchronous subsystems) and that only the temporal behavior is learned.

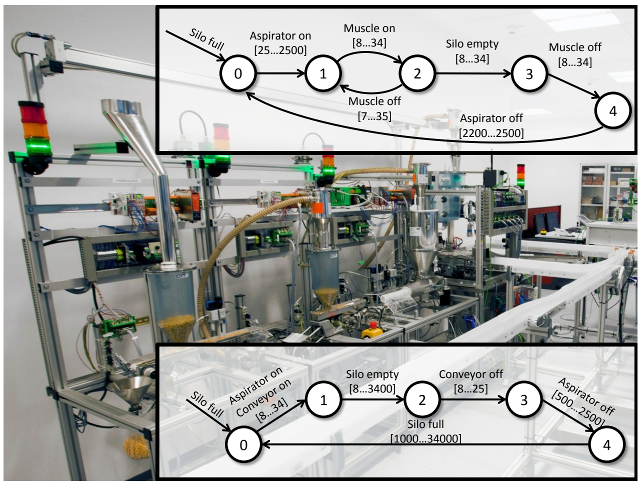

Figure 3: Learned Timed Automata for a manufacturing plant.

The data acquisition for this solution (I/F 1 in Figure 2) has been implemented using a direct capturing of Profinet signals including an IEEE 1588 time-synchronization. The data is offered via OPC UA (I/F 2). On the learning layer, timed automata are learned from historical data and compared to the observed behavior. Also, the sequential behavior of the observed events as well as the timing behavior is checked, anomalies are signaled via I/F 3. On the conceptual layer it is decided whether an anomaly is relevant. Finally, a graphical user interface is connected to the conceptual layer via OPC UA (I/F 4).

Figure 3 shows learned automata for a manufacturing plant: The models correspond to modules of the plants, transitions are triggered by a control signals and are annotated with a learned timing interval.

4.2 Energy Analysis In Process Industry

Analyzing the energy consumption in production plants has some special challenges: Unlike the discrete systems described in Section 4.1, also continuous signals such as the energy consumption must be learned and analyzed. But also the discrete signals must be taken into consideration because continuous signals can only be interpreted with respect to the current system's status, e.g. it is crucial to know whether a valve is open or whether a robot is turned on. And the system's status is usually defined by the history of discrete control signals.

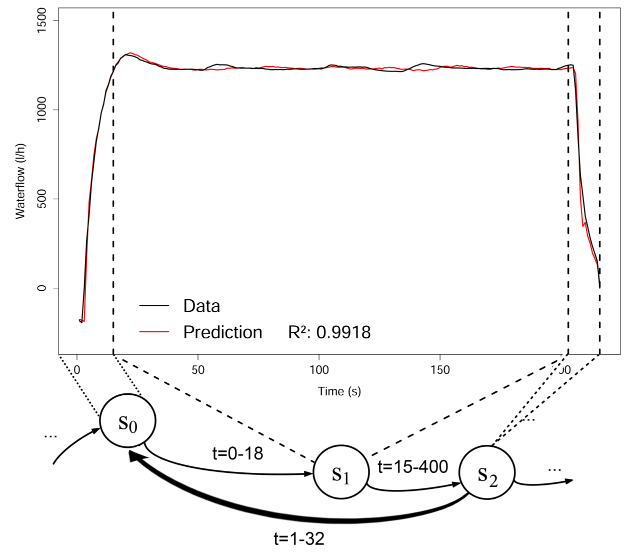

Figure 4: A learned hybrid automaton modeling a pump.

In [19], an energy anomaly detection system is described which analyzes three production plants. Ethercat and Profinet is used for I/F 1 and OPC UA for I/F 2. The collected data is then condensed on the learning layer into hybrid timed automata. Also on this layer, the current energy consumption is compared to the energy prediction. Anomalies in the continuous variables are signaled to the user via mobile platforms using web services (I/F 3 and 4).

In Figure 4, a pump is modeled by means of such automata using the flow rate and switching signals. The three states $S0$ to $S2$ are separating the continuous function into three linear pieces which can then be learned automatically.

Figure 5 shows a typical learned energy consumption (here for bulk good production).

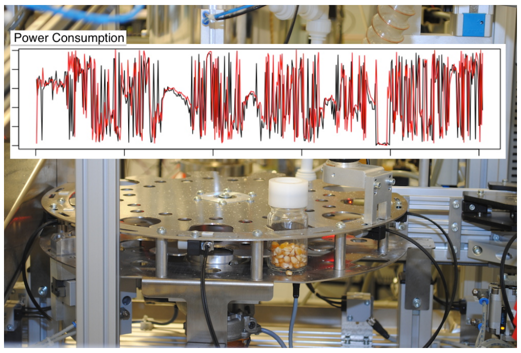

Figure 5: A measured (black line) and a learned power consumption (red line).

4.3 Big Data Analysis in Manufacturing Systems

Analyzing historical process data during the whole production cycle requires new architectures and platforms for handling the enormous volume, variety and velocity of the data. Data analysis pushes the classical data acquisition and storage up to its limits, i.e. big data platforms are need.

In the assembling line of the SmartFactoryOWL, a small factory used for production and research, a big data platform is established to acquire, store and visualize the data from the production cycles. In Figure 6 the architecture of the big data platform is depicted.

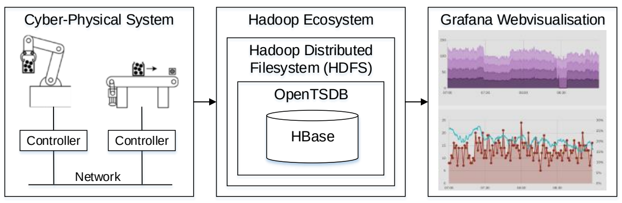

Figure 6: Data Analysis Plattform in Manufacturing

The CPS is connected through OPC UA (I/F 1 in Figure 2) with an Hadoop ecosystem. Hadoop itself is an software framework for scalable distributed computing. The process data is stored in an non-relational database (HBase) which is based on a distributed file-system (HDFS). On top of HBase, a time-series database $OpenTSDB$ is used as an interface to explore and analyze the data (I/F 2 in Figure 2). Through this database it is possible to do simple statistics such as mean-values, sums or differences, which is usually not possible within the non relational data stores.

Using the interfaces of OpenTSDB or Hadoop, it becomes possible to analyze the data directly on the storage system. Hence, the volume of a historical dataset need not be loaded into a single computer system, instead the algorithms can work distributively on the data. A web interface can be used to visualize the data as well as the computed results. In Figure 6, grafana is used for data visualization. In the SmartFactoryOWL this big data platform is currently being connected to the application scenarios from Sections 4.1 and 4.2.

4.4 Anomaly Detection in Aircraft Flight Data

Fault detection and isolation schemes are designed to detect the onset of adverse events during operations of complex systems, such as aircraft and industrial processes. In other work, we have discussed approaches using machine learning classifier techniques to improve the diagnostic accuracy of the online reasoner on board of the aircraft [20]. In this paper, we discuss an anomaly detection method to find previously undetected faults in aircraft system [21].

The flight data used for improving detection of existing faults and discovering new faults was provided by Honeywell Aerospace and recorded from a former regional airline that operated a fleet of 4-engine aircraft, primarily in the Midwest region of the United States. Each plane in the fleet flew approximately 5 flights a day and data from about 37 aircraft was collected over a five year period. This produced over 60,000 flights. Since the airline was a regional carrier, most flight durations were between 30 and 90 minutes. For each flight, 182 features were recorded at sample rates that varied from 1Hz to 16Hz. Overall this produced about 0.7 TB of data.

Situations may occur during flight operations, where the aircraft operates in previously unknown modes that could be attributed to the equipment, the human operators, or environmental conditions (e.g., the weather). In such situations, data-driven anomaly detection methods [12], i.e., finding patterns in the operations data of the system that were not expected before can be applied. Sometimes, anomalies may represent truly aberrant, undesirable and faulty behavior; however, in other situations they may represent behaviors that are just unexpected. We have developed unsupervised learning or clustering methods for off-line detection of anomalous situations. Once detected and analyzed, relevant information is presented to human experts and mission controllers to interpret and classify the anomalies.

Figure 7 illustrates our approach. We started with curated raw flight data (layer "Big Data Platform" in Figure 2), transforming the time series data associated with the different flight parameters to a compressed vector form using wavelet transforms. The next step included building a dissimilarity matrix of pairwise flight segments using the Euclidean distance measure, followed by a subsequent step where the pairwise between flight distances was used to run a 'complete link' hierarchical clustering algorithm [22] (layer "Learning" in Figure 2). Run on the flight data, the algorithm produced a number of large clusters that we considered to represent nominal flights, and a number of smaller clusters and outlier flights that we initially labeled as anomalous. By studying the feature value differences between the larger nominal and smaller anomalous clusters with the help of domain experts, we were able to interpret and explain the anomalous nature ("Conceptual Layer" in Figure 2).

These anomalies or faults represented situations that the experts had not considered before; therefore, this unsupervised or semi-supervised data driven approach provided a mechanism for learning new knowledge about unanticipated system behaviors. For example, when analyzing the aircraft data, we found a number of anomalous clusters. One of them turned out to be situations where one of the four engines of the aircraft was inoperative. On further study of additional features, the experts concluded that these were test flights conducted to test aspects of the aircraft, and, therefore, they repesented known situations, and, therefore, not an interesting anomaly. A second group of flights were interpreted to be take offs, where the engine power was set much higher than most flights in the same take off condition. Further analysis of environmental features related to these set of take-off's revealed that these were take-offs from a high altitude airport at 7900 feet above sea level.

A third cluster provided a more interesting situation. The experts when checking on the features that had significantly different values from the nominal flights realized that the auto throttle disengaged in the middle of the aircraft's climb trajectory. The automatic throttle is designed to maintain either constant speed during takeoff or constant thrust for other modes of flight. This was an unusual situation where the auto thruster switched from maintaining speed for a takeoff to a setting that applied constant thrust, implying that the aircraft was on the verge of a stall. This situation was verified by the flight path acceleration sensor shown in Figure 7. By further analysis, the experts determined that in such situations the automatic throttle would switch to a possibly lower thrust setting to level the aircraft and compensate for the loss in velocity. By examining the engine parameters, the expert verified that all the engines responded in an appropriate fashion to this throttle command. Whereas this analysis did not lead to a definitive conclusion other than the fact the auto throttle, and therefore, the aircraft equipment, responded correctly, the expert determined that further analysis was required to answer the question *"why did the aircraft accelerate in such a fashion and come so close to a stall condition?"*. One initial hypothesis to explain these situations was pilot error.

4.5 Reliability and Fault Tolerant Control

Most complex CPSs are safety-critical systems that operate with humans-in-the-loop. In addition to equipment degradation and faults, humans can also introduce erroneous decisions, which becomes a new source of failure in the system. Figure 8 represents possible faults and cyber-attacks that can occur in a CPS.

There are several model-based fault tolerant control strategies for dynamic systems in the literature (see for example [23] and [24]). Research has also been conducted to address network security and robust network control problems (see for example [25] and [26]). However, these methods need mathematical models of the system, which may not exist for large scale complex systems. Therefore, data driven control [27] and data driven fault tolerant control [28] have become an important research topic in recent years. For CPSs, there are more aspects of the problem that need to be considered. As it is shown in Figure 8, there are many sources of failure in these systems.

We propose a hybrid approach that uses an abstract model of the complex system and utilizes the data to ensure the compatibility between model and the complex system. Data abstraction and machine learning techniques are employed to extract patterns between different control configurations and system outputs unit by computing the correlation between control signals and the physical subsystems outputs. The highly correlated subsystems (layer "Learning" in Figure 2) become candidates for further study of the effects of failure and degradation at the boundary of these interacting subsystems. For complex systems, all possible inteeractions and their consequences are hard to pre-determine, and data-driven approaches help fill this gap in knowledge to support more informed decision-making and control. A case-based reasoning module can be designed to provide input on past successes and failed opportunities, which can then be translated by human experts into operational monitoring, fault diagnosis, and control situations ('Conceptual Layer" in Figure 2). Some of the control paradigms that govern appropriate control configurations, such as modifying sequence of mission tasks and switching between different objectives or changing the controller parameters (layer Adaptation in Figure 2) are being studied in a number of labs including ours [29].

Example Fault Tolerant Control of Fuel Transfer System The fuel system supplies fuel to the aircraft engines. Each individual mission will have its own set of requirements. However, common requirements such as saving the aircraft Center of Gravity (CG), safety, and system reliability are always critical. A set of sensors included in the system to measure different system variables such as the fuel quantity contained in each tank, engines fuel flow rates, boost pump pressures, position of the valves and etc.

There are several failure modes such as the total loss or degradation in the electrical pumps or a leakage in the tanks or connecting pipes in the system. Using the data and the abstract model we can detect and isolate the fault and estimate its parameters. Then based on the type fault and its severity the system reconfiguration unit chooses the proper control scenario form the control library. For example in normal situation the transfer pumps and valves are controlled to maintain a transfer sequence to keep the aircraft center of gravity within limits. This control includes maintaining a balance between the left and right sides of the aircraft. When there

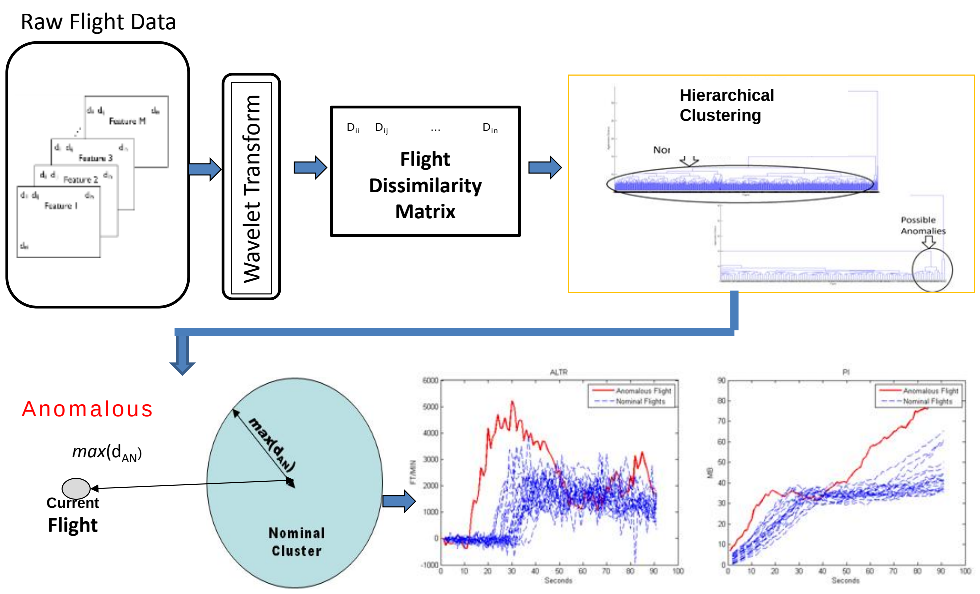

Figure 7: Data Driven Anomaly Detection Approach for Aircraft Flights

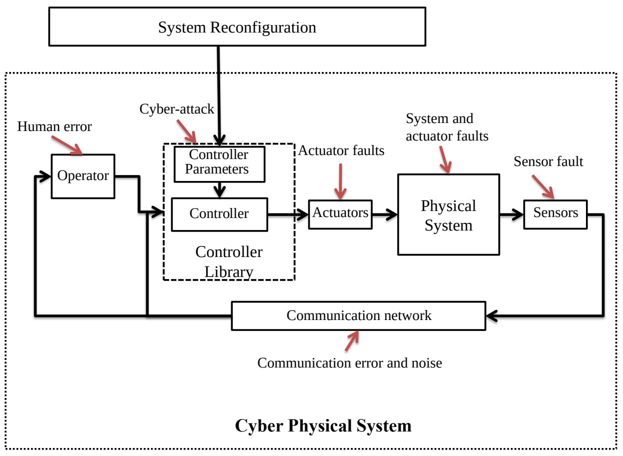

Figure 8: Possible faults in a CPS.

is a small leak, normally the system can tolerate it depending on where the leak is, but the leak usually grows over time. Therefore we need to estimate the leakage rate and reconfigure the system to move the fuel from the tank or close the pipe before critical situation.

5 Conclusions

Data-driven approaches to the analysis and diagnosis of Cyber-Physical Systems (CPSs) are always inferior to classical model-based approaches, where models are created manually by experts: Experts have background knowledge which can not be learned from models and experts automatically think about a larger set of system scenarios than can be observed during a system's normal lifetime.

So the question is not whether data-driven or expert-driven approaches are superior. The question is rather which kind of models can we realistically expect to exist in real-world applications—and which kind of models must therefore be learned automatically. This becomes especially important in the context of CPSs since these systems adapt themselves to their environment and show therefore a changing behavior, i.e. models would also have be adapted frequently.

In Sections 4.1 and 4.2, structural information about the plant is imported from the engineering chain and the temporal behavior is learned in form of timed automata. In Section 4.5, an abstract system model describing the input/output structure and the main failure types is provided and again the behavior is learned. These approaches are typical because in most applications structural information can be gained from earlier engineer phases while behavior models hardly exist and are almost never validated with the real system.

Looking at the learning phase, the first thing to notice is that all described approaches work and deliver good results: For CPSs, data-driven approaches have moved into the focus of research and industry. And they are well suited for CPSs: They adjust automatically to new system configurations, they do not need manual engineering efforts and they make usage of the now available large number of data signals—connectivity being a typical feature of CPSs.

Another common denominator of the described application examples is that the focus is on anomaly detection rather than on root cause analysis: for data-driven approaches it is easier to learn a model of the normal behavior than learning erroneous behavior. And it is also typical that the only root cause analysis uses a case-based approach (Section 4.5), case-based approaches being suitable for data-driven solutions to diagnosis.

Finally, the examples show that the proposed cognitive architecture (Figure 2) matches the given examples:
Big Data Platform: Only a few examples (e.g. Section 4.3) make usage of explicit big data platforms, so-far solutions often use proprietary solutions. But with the growing size of the data involved, new platforms for storing and processing the data are needed.
Learning: All examples employ machine learning technologies—with a clear focus on unsupervised learning techniques which require no a-priori knowledge such as clustering (Section 4.4) or automata identification (Sections 4.1, 4.2).
Conceptual Layer: In all examples, the learned models are evaluated on a conceptual or symbolic level: In Section 4.4, clusters are compared to new observations and data-cluster

distances are used for decision making. In Sections 4.1 and 4.2, model predictions are compared to observations. And again, derivations are decided on by a conceptual layer.
Task-Specific HMI: None of the given examples works completely automatically, in all cases the user is involved in the decision making.
Adaption: In most cases, reactions to detected problems are up to the expert. The use case from Section 4.5 is an example for an automatic reaction and the usage of analysis results for the control mechanism.

Using such a cognitive architecture would bring several benefits to the community: First of all, algorithms and technologies in the different layers can be changed quickly and can be re-used. E.g. learning algorithms from one application field can be put on top of different big data platforms. Furthermore, currently most existing approaches mix the different layers, making the comparison of approaches to the analysis of CPSs difficult. Finally, such an architecture helps to clearly identify open issues for the development of smart monitoring systems.

Acknowledgments The work was partly supported by the German Federal Ministry of Education and Research (BMBF) under the project "Semantics4Automation" (funding code: 03FH020I3), under the project "Analyse großer Datenmengen in Verarbeitungsprozessen (AGATA)" (funding code: 01IS14008 A-F) and by NASA NRA NNL09AA08B from the Aviation Safety program. We also acknowledges the contributions of Daniel Mack, Dinkar Mylaraswamy, and Raj Bharadwaj on the aircraft fault diagnosis work.

References

[1] E.A. Lee. Cyber physical systems: Design challenges. In *Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on*, pages 363–369, 2008.

[2] Ragunathan (Raj) Rajkumar, Insup Lee, Lui Sha, and John Stankovic. Cyber-physical systems: The next computing revolution. In *Proceedings of the 47th Design Automation Conference*, DAC '10, pages 731–736, New York, NY, USA, 2010. ACM.

[3] Peter C. Evans and Marco Annunziata. Industrial internet: Pushing the boundaries of minds and machines. Technical report, GE, 2012.

[4] Promotorengruppe Kommunikation. Im fokus: Das industrieprojekt industrie 4.0, handlungsempfehlungen zur umsetzung. Forschungsunion Wirtschaft-Wissenschaft, March 2013.

[5] L. Christiansen, A. Fay, B. Opgenoorth, and J. Neidig. Improved diagnosis by combining structural and process knowledge. In *Emerging Technologies Factory Automation (ETFA), 2011 IEEE 16th Conference on*, Sept 2011.

[6] Rolf Isermann. Model-based fault detection and diagnosis - status and applications. In *16th IFAC Symposium on Automatic Control in Aerospace*, St. Petersbug, Russia, 2004.

[7] Johan de Kleer, Bill Janssen, Daniel G. Bobrow, Tolga Kurtoglu Bhaskar Saha, Nicholas R. Moore, and Saravan Sutharshana. Fault augmented modelica models. *The 24th International Workshop on Principles of Diagnosis*, pages 71–78, 2013.

[8] D. Klar, M. Huhn, and J. Gruhser. Symptom propagation and transformation analysis: A pragmatic model for system-level diagnosis of large automation systems. In *Emerging Technologies Factory Automation (ETFA), 2011 IEEE 16th Conference on*, pages 1–9, Sept 2011.

[9] GE. The rise of big data - leveraging large time-series data sets to drive innovation, competitiveness and growth - capitalizing on the big data oppurtunity. Technical report, General Electric Intelligent Platforms, 2012.

[10] A. Katasonov, O. Kaykova, O. Khriyenko, S. Nikitin, and V. Terziyan. Smart semantic middleware for the internet of things. In *5th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2008.

[11] Michael Stonebraker. Sql databases v. nosql databases. *Communications of the ACM*, 53(4):10–11, 2010.

[12] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41.3:1–72, Sept 2009.

[13] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I Stoica. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, page 10, June 2010.

[14] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *Proceedings 26th IEEE Symposium on Mass Storage Systems and Technologies (MSST)*, pages 1–10, May 2010.

[15] M JAYASREE. Data mining: Exploring big data using hadoop and mapreduce. *International Journal of Engineering Sciences Research-IJESR*, 4(1), 2013.

[16] Friedhelm Nachreiner, Peter Nickel, and Inga Meyer. Human factors in process control systems: The design of human–machine interfaces. *Safety Science*, 44(1):5–26, 2006.

[17] Sicco Verwer. *Efficient Identification of Timed Automata: Theory and Practice*. PhD thesis, Delft University of Technology, 2010.

[18] Oliver Niggemann, Benno Stein, Asmir Vodenčarević, Alexander Maier, and Hans Kleine Büning. Learning behavior models for hybrid timed systems. In *Twenty-Sixth Conference on Artificial Intelligence (AAAI-12)*, pages 1083–1090, Toronto, Ontario, Canada, 2012.

[19] Bjoern Kroll, David Schaffranek, Sebastian Schriegel, and Oliver Niggemann. System modeling based on machine learning for anomaly detection and predictive maintenance in industrial plants. In *19th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep 2014.

[20] D.L.C. Mack, G. Biswas, X. Koutsoukos, and D. Mylaraswamy. Learning bayesian network structures to augment aircraft diagnostic reference model, "to appear". *IEEE Transactions on Automation Science and Engineering*, 17:447–474, 2015.

[21] Daniel LC Mack. *Anomaly Detection from Complex Temporal Spaces in Large Data*. PhD thesis, Vanderbilt University, Nashville, TN. USA, 2013.

[22] Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[23] Jiang Jin. Fault tolerant control systems - an introductory overview. *Acta Automatica Sinica*, 31(1):161–174, 2005.

[24] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki. *Diagnosis and fault-tolerant control*. Springer-Verlag, Sep 2003.

[25] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. S. Sastry. Foundations of control and estimation over lossy networks. In *In Proceedings of the IEEE*, volume 95, pages 163 – 187, Jan 2007.

[26] B. Schneier. Security monitoring: Network security for the 21st century. In *Computers Security*, 2001.

[27] Zhong-Sheng Hou and Zhuo Wang. From model-based control to data-driven control: Survey, classification and perspective. *Information Sciences*, 235:3–35, 2013.

[28] Hongm Wang, Tian-You Chai, Jin-Liang Ding, and Martin Brown. Data driven fault diagnosis and fault tolerant control: some advances and possible new directions. *Acta Automatica Sinica*, 25(6):739–747, 2009.

[29] Z. S. Hou and J. X. Xu. On data-driven control theory: the state of the art and perspective. *Acta Automatica Sinica*, 35:650–667, 2009.