Real-Time Monitoring and Long-Term Analysis by Means of Embedded Systems

Tino Noack
Supervised by Prof. Ingo Schmitt

TU Cottbus
Institute of Computer Science, Information and Media Technology
Chair of Database and Information Systems
Tino.Noack@tu-cottbus.de

Abstract. This paper sketches an interdisciplinary doctoral research. The main contribution is amongst others the combination of existing approaches for real-time monitoring and long-term analysis. This includes data stream management, event condition action rules, complex event processing as well as data mining technologies. As a practical use case we introduce briefly a scenario related to the failure management system of the International Space Station Columbus Module. Our research is based on three main assumptions and we identify five monitoring requirements. Furthermore, we describe a system model that is known as the state space. Here, the state space represents the knowledge about the monitored target system. Additionally, we present a cyclic monitoring process chain that represents a dynamic and flexible monitoring approach. Our proposed monitoring architecture respects the complexity of system monitoring as well as today's and future monitoring requirements.

Keywords: Monitoring, Real-Time, Long-Term, Embedded System, Data Stream Management, Data Mining, Complex Event Processing

1 Introduction

Embedded systems are widely used in today's products such as cars, trains, airplanes or spacecrafts, where they are often used for controlling and monitoring purposes. In most of cases, these products are subject to real-time requirements and reliability. Nowadays, monitoring technical systems is a widespread research area and it is applied in many heterogeneous application domains. While monitoring solutions are often designed, developed and implemented for specific applications, production costs increases and at the same time, flexibility of the monitoring solution is getting more and more lost because of increasing complexity. Significant applications can be found, for instance, in the area of spacecraft monitoring ([17], [18]). Spacecraft monitoring is very challenging because complete system tests in the latter application environment (the space) and continual maintenance are impracticable respectively impossible.

Because of the increasing complexity of today's products improved monitoring approaches are needed that respect today's and future requirements. This paper sketches an interdisciplinary doctoral research. We aim to research on the combination of existing, well known and well applied approaches that can be adequately used for combining real-time monitoring and long-term analysis of events by means of embedded systems [19]. Due to the use of existing approaches it is consequently possible to reduce production costs. Our research includes data stream management [1], event condition action (ECA) [11] rules, data mining technologies [23] as well as complex event processing (CEP) [13]. Due to the complexity of system monitoring a dynamic and flexible monitoring approach is proposed here.

Our monitoring approach is based on the following three assumptions:

1. Across different application domains the underling monitoring methodologies and algorithms are similar.
2. It is impossible to exclude any occurrence of errors during run-time. Thus, any change of the system behaviour must be adequately followed by an appropriate action.
3. The monitoring process is semi-automatic. Information technologies are used to facilitate the monitoring process.

The rest of the paper is organized as follows. In Section 2 we describe briefly a use case that is related to the failure management system of the ISS Columbus Module. Section 3 defines the term embedded system as we intend to use it in our research. Section 4 summarizes monitoring requirements and based on this, Section 5 delineates our research questions. Section 6 describes the system model that we intend to use for the suggested monitoring approach. Section 7 details our approach. Our contribution is amongst others the combination of existing, well known and well applied approaches for the combination of real-time monitoring and long-term analysis. Section 8 summarizes existing solutions and finally, a conclusion is given in section 9.

2 Use Case: Long-Term Degradation of ISS Columbus Inter Module Ventilation Return Fan Assembly

The ISS Columbus **I**nter Module Ventilation **R**eturn **F**an **A**ssembly (IRFA) [17] is used to provide air circulation. The IRFA air circulation is necessary to prevent dead air pockets, for smoke detection (fire), cabin heat collection and for air revitalisation. Irregularities of the IRFA arouse because of long-term wearout effects (e.g. bearing wearout). Figure 1 depicts the collected measurements. The attribute *Pressure* describes the pressure haed that is generated by the IRFA, the attribute *IRFA Speed* describes the speed of the fan and the attribute *Input Current* describes the incoming electrical current. The Input Current is equivalent to the produced air flow and to the mechanical friction losses. The uppermost diagram of Figure 1 shows long-term wearout effects (I.) and the undermost one shows short-term influencing factors respectively the irregularities (II.). The bearing wearout (I.) led to a continuous increasing Input Current from the beginning of February to the beginning of April whereas the Pressure and the IRFA Speed are untainted. The IRFA irregularities occurred on the day 106 in 2008 (II.). The failure event led to erratic IRFA Speed and consequently to erratic air flow. The failure event lasted 210 seconds. There are two implementations for automatic failure detection and deactivation of the IRFA. But none of them covered the unknown failure signature. The failure event was manually detected and manually recovered by the flight control team instead of automatic detection. In worse cases those failure situations could remain for a long time period without recognition.

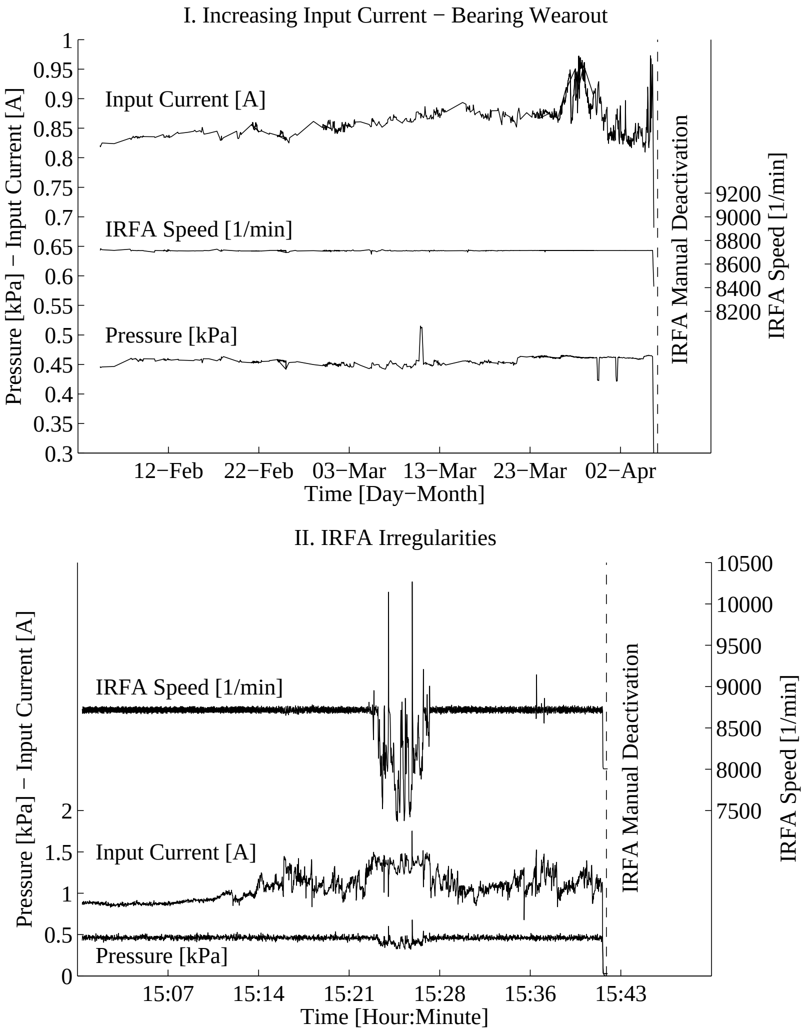

Fig. 1. Increasing Input Current (I.) and Irregularities (II.) of ISS Columbus IRFA

3 Introduction to Embedded Systems

Figure 2 sketches an abstract architecture of an embedded system. Embedded systems are embedded into a product. The product is embedded into the product environment. Embedded systems consist of electronic assemblies (hardware) that represent the system components. Additionally, these electronic assemblies are equipped with software. Embedded systems are subject to resource restrictions such as processor speed, power and memory consumption. The embedded system interacts with the product and the product environment via sensors and actuators. The electronic assemblies can be connected by an internal network. Furthermore, the embedded system can be temporarily connected to an external information system via an external network. More information about embedded systems can be found amongst others in [16], [20] and [24].

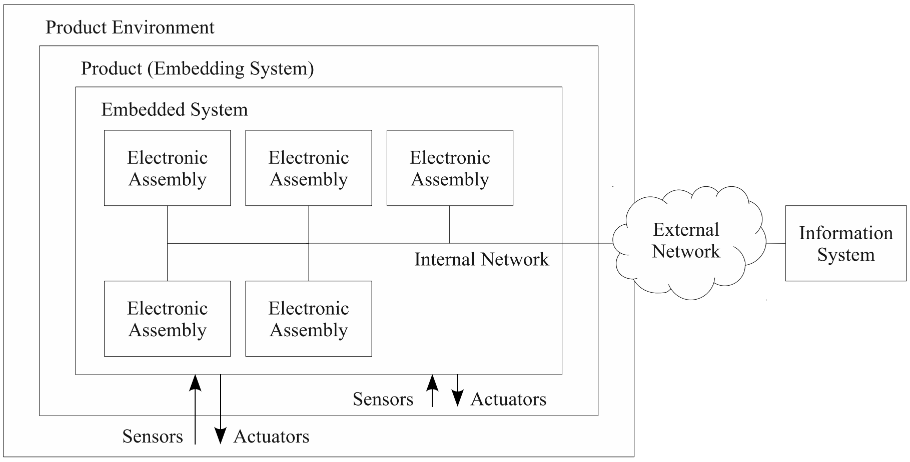

Fig. 2. Abstract Architecture of an Embedded System [19]

4 Monitoring Requirements

According to the presented use case and considering the abstract architecture of an embedded System it is possible to describe monitoring requirements. This involves the following five dimensions: time, locality, knowledge, system resources and sharpness. Figure 3 depicts the mentioned requirements.

Time: This requirement refers to the temporal and continual changing of the system components.

- *Short-Term*: Abrupt changes can occur (e.g. collision). It is needed to detect such abrupt changes in real-time.
- *Long-Term*: In order to detect long-term influencing factors and changes (e.g. wear and tear) long-term analysis is required.

Locality: This requirement refers to interrelation effects of influencing factors and the spatial location of monitoring.

– *Local*: Failures that relate on few system components must be detected by means of local monitoring.
– *Global*: Because of the rising complexity of today's products the correlation of influencing factors increases. Thus, complex interrelations arise between system components. It is needed to gather and to detect such complex interrelations by the use of global analysis.

Knowledge: This requirement refers to the available information about the embedded system, the product and its environment.

– *Known*: It is necessary to employ knowledge about the embedded system, the product and its environment as comprehensive and goal-oriented as possible for the monitoring process.
– *Unknown*: Because of unknown and unforeseeable conditions a dynamic, flexible and adaptable monitoring process is needed.

System Resources: This requirement refers to all available resources for the monitoring process.

– *Unrestricted*: Monitoring in particular long-term monitoring requires extremely many system resources. From this point of view a combination of internal and external monitoring resources is needed (hybrid monitoring [22]).
– *Restricted*: Because of restricted system resources of embedded systems it is necessary to use them adequately and goal-oriented for the internal monitoring process.

Sharpness: This requirement refers to the interpretation respectively the processing of conditions ([5], [21]).

– *Crisp*: System states must be detected exactly and reliably by the use of binary processing (Boolean logic). For example, if a threshold value is reached.
– *Non-Crisp*: For particular problems crisp processing is inadequate. From this point of view it necessary to generalize binary processing by means of affiliation degrees between 0 and 1. The value 1 implies full affiliation and the value 0 implies the opposite.

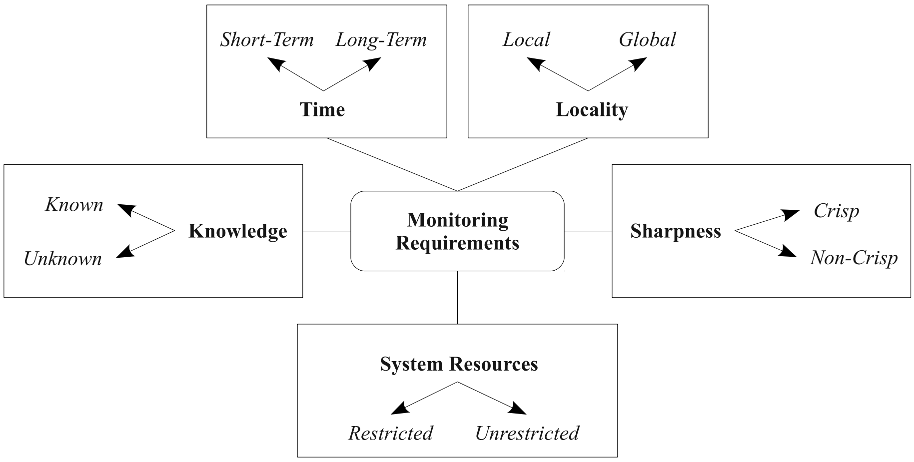

Fig. 3. Monitoring Requirements [19]

5 Research Questions

There is a gap between real-time monitoring and long-term analysis of events which affect the reliability of the system. Therefore we aim to research on the combination of real-time monitoring and long-term analysis of events. In a first step we consider all requirements excepting sharpness. Figure 4 summarizes the research question.

Long-term analysis needs usually a huge amount of processing resources. Hence, it must be processed offline and on an external information system with nearly unrestricted system resources. Furthermore, data mining technologies are semi-automatic. Thus, specialized staff is needed that observes and fosters the data mining process. Data mining technologies are applied here to learn classifiers. These classifiers are represented by means of ECA rules. The persistent stored data gives a global view to the whole system. It can be used to identify relevant interrelations. We use data mining technologies to increase knowledge about the system over time.

With respect to the above-mentioned use case the data is gathered on an external information system. This persistent stored data is used to learn classifiers that can distinguish between normal, abnormal and anomalous behaviour of the IRFA, known as anomaly detection [9]. Furthermore, the persistent stored data can be used to detect the gradual change of system components over a long time period. This helps to identify long-term influencing effects of wear and tear.

Real-time monitoring must be processed on the embedded system that is subject to resource restrictions. Monitoring must be processed automatically, online and without any user interactions. Changes of the system behaviour that require immediate responses must be detected adequately with respect to real-time requirements. Here, the learned classifiers respectively the ECA rules are transferred to the embedded system and afterwards applied for behaviour change and anomaly detection. Here, CEP is a selected tool to utilize the ECA rules onto continuous data streams. ECA rules represent the knowledge about the system. Behaviour that do not fit to this rules might be labelled as anomalous. This is a local point of view because only a subset of attributes is used to define rules for specific behaviour.

With respect to the above-mentioned use case the irregularities of the IRFA has entailed a significant and abrupt change of the system behaviour. The ECA approach is described subsequently. Here, an event is the behaviour of the system at a specific time. The condition refers to the learned classifiers respectively to the rules that are used to classify the behaviour of the system at a specific time. An action could be a failure massage or the automatic deactivation of the IRFA to avoid material damage.

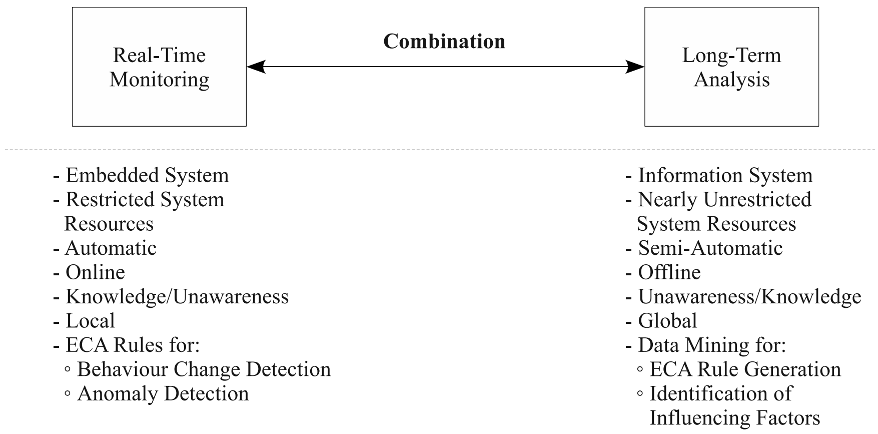

Fig. 4. Combination of Real-Time Monitoring and Long-Term Analysis [19]

6 The Model

A key issue is the understanding of the input data. Sensors produce continuous data. These continuous sensor data can be construed as data streams. A data stream consists of a sequence of data items. Often, this sequence is very large. A system that processes data streams has no a priori control about the order of arriving data items. A renewed transmission of lost data items is impossible. More information about data streams and data stream processing can be found amongst others in [1], [3], [8] and [14].

We construe a set of features as a set of attributes $A_1, \dots, A_n$ that represent the state variables of the target system. They can be amongst others nominal, ordinal or metrical. Attribute values are functions on time i.e. values of A_i are values of $a_i : T \rightarrow \mathbb{R}$ where T is a time representation and $\mathbb{R}$ is the set of real numbers.

Therefore, a state at time t is represented as a state vector

$$\vec{a}(t) = \begin{pmatrix} a_1(t) \\ a_2(t) \\ \vdots \\ a_n(t) \end{pmatrix}.$$

The space that is spanned by the attributes is called the state space. The number of attributes defines the number of dimensions of the state space. A set of state vectors in the state space that represents similar kinds of states can be geometrically interpreted. This geometrical interpretation is known as a cluster in the area of data mining technologies ([23], [9], [6], [2]).

Figure 5 depicts the state space in a time frame of a system considering two attributes A_1 and A_2. For better clarity, Figure 5 is incomplete and the state vectors are represented by means of dots. Let S be the set of all possible system states respectively the state space, let S_k be the set of known system states and let S_u be the set of unknown system states such that $S_k \bigcup S_u = S$ and $S_k \bigcap S_u = \varnothing$. Hence, unknown system states are complementary to known system states. The clusters C_{k1} and C_{k2} of Figure 5 are representing sets of known system states. The cluster C_{u1} and the points p_{u1} and p_{u2} are exemplary for unknown system states. In [9] these unknown system states are called anomalies. The aim of the learned classifiers is to classify at each time t a state vector to a known cluster or to label it as unknown respectively as anomalous. Hence, the ECA rules are used for classification purposes respectively supervised learning and they represent the classifiers that were learned by means of data mining technologies.

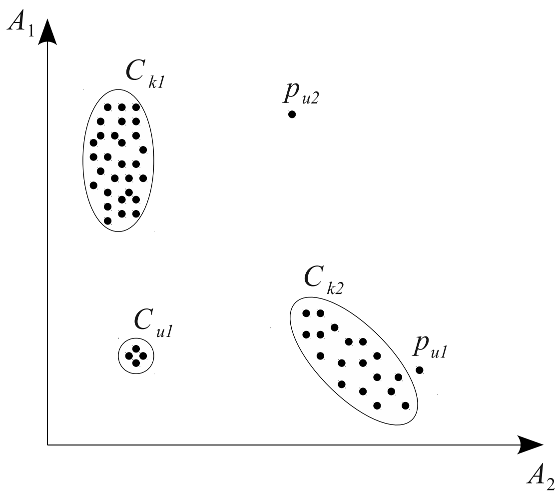

Fig. 5. State Space [9]

7 Combination of Real-Time Monitoring and Long-Term Analysis

As already described, the focus of interest lies in the combination of real-time monitoring and long-term analysis. The aim is to learn a model respectively a system state space that represents the knowledge of the target system that is monitored. In the beginning, this section describes a cyclic monitoring process chain. Then, this monitoring process chain is mapped to an abstract monitoring architecture.

The monitoring process chain is depicted in Figure 6. It is divided into real-time monitoring that takes place on the embedded system and in long-term analysis that takes place on an external information system.

The monitoring process chain starts with events. From the CEP point of view each state vector is construed as an event. Pre-processing is the second step. It can be used amongst others for noise reduction, relevant event selection and for windowing to minimize processing efforts. Rule execution is the third step and it is used to perform previously defined rules onto the pre-processed events. The fourth step can be used to send messages to actuators. The fifth step is used for temporal storage mechanisms. This involves data aggregation to minimize data volume as well as selected storage strategies such as ring buffers or embedded databases. The smaller cyclic arrow indicates that these steps are separated from long-term analysis that starts with the following step. The sixth step is the data transmission to the external information system. Because of uncertainty of the external network data can only be transmitted from time to time when the communication path is available. Each part of these steps should be interchangeable and configurable (like plug-ins) to provide a dynamic and flexible monitoring solution. From this point of view it is possible to tailor the CEP engine by means of plug-ins to the underling hardware and to the intended monitored approach. The seventh step is loading of the received data into a persistent storage like a data warehouse (DWH). The eighth step is used for rule generation by means of data mining technologies. Presently, this includes the following classification respectively supervised learning strategies: rule induction, support vector machine and nearest neighbour. Mostly, these selected data mining technologies must also be combined for appropriate classification ([9], [23]). The ninth step is used to evaluate new generated rules and to compare them with already applied rules to avoid side effects. The last step is the transmission of new knowledge to the embedded system. This involves the accommodation and reconfiguration based on new knowledge of the applied monitoring system. Steps one to six should be automatic and steps seven to ten are semi-automatic and must be observed by specialized staff. The entire monitoring process chain is cycling to increase the knowledge over time about the target system that is monitored.

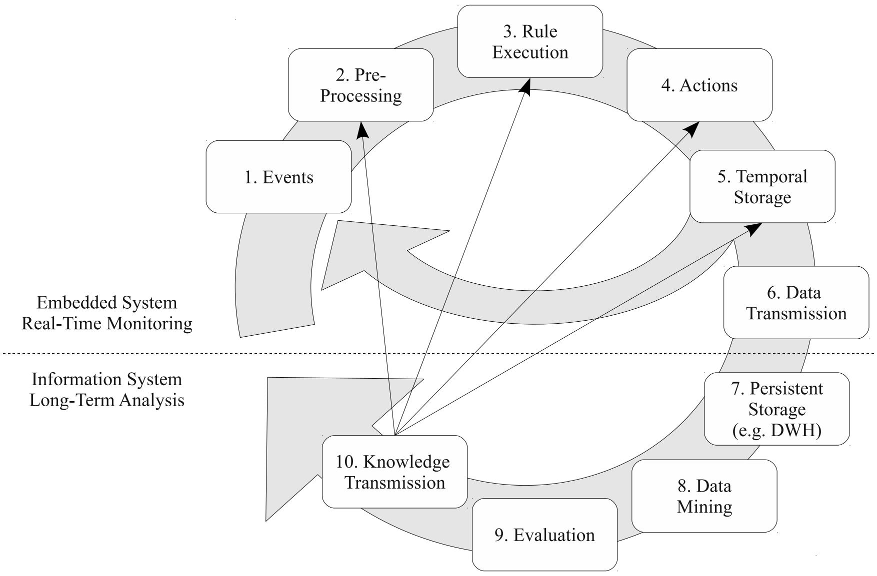

Fig. 6. Cyclic Monitoring Process Chain

The proposed monitoring architecture is depicted in Figure 7. It is based on the mentioned monitoring process chain. Sensors produce continuous data streams that are transmitted via the internal network. These events respectively the state vectors need to be computed continuously and with respect to real-time requirements by means of the CEP engine. The CEP engine has to produce according to the action part of the applied ECA rules actions. Furthermore, the data stream is aggregated and temporarily stored before it is transmitted to the external information system. The external information system is used for long-term analysis to derive new rules and for refinement of existing rules. Afterwards, these rules have to be evaluated and need to be transferred to the embedded system.

8 Existing Solutions

Data stream management systems (DSMS) such as STREAM [1] or Aurora [7] are used for processing and exploring data streams. Especially Aurora consists of a box and arrow architecture model such as a plug-in system. An overview of DSMS is given in [14]. CEP engines such as CAYUGA [10] or ESPER [12] are used to process rules onto data streams by means of query languages. These query languages can potentially used for ECA rule definition. An overview of CEP engines is given in [13]. However, the mentioned systems were not intended for monitoring approaches by means of data mining technologies.

VEDAS [15] reflects a cup of the above-mentioned monitoring requirements. The detection of unusual patterns of driving characteristics is one of the main objectives

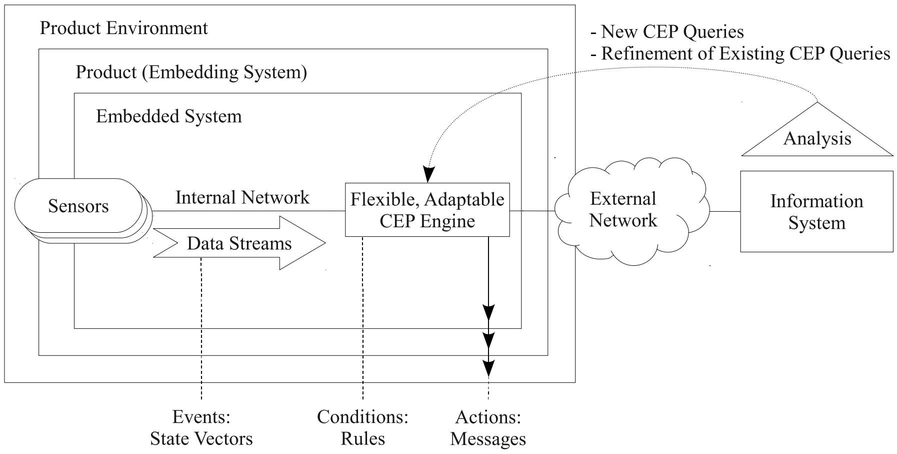

Fig. 7. Suggested Monitoring Approach [19]

of VEDAS. As we suggested existing data mining technologies are used. The difference lies in the usage of unsupervised data stream mining technologies. Supervised technologies are not support by VEDAS. Furthermore, VEDAS is not laid out for processing rules onto data streams. It lacks a strict separation between real-time monitoring and long-term analysis as well as automatic and semi-automatic functionalities.

Odysseus [4] is a very young research project. Odysseus is called data stream management framework and it is based on a service-oriented architecture. It should enable the evaluation of heterogeneous algorithms and approaches in the research area of CEP. Especially this service-oriented architecture makes Odysseus very worthwhile for the evaluation of our suggested monitoring approach.

9 Conclusion

There is a need for new monitoring solutions that respect today's and future requirements. This paper sketches an interdisciplinary doctoral research. The main contribution is amongst others the combination of real-time monitoring and long-term analysis by means of embedded systems, data stream management, data mining technologies, ECA rules and CEP. The suggested approach is based on three assumptions. Additionally, five monitoring requirements were identified here. The analysis of existing solutions pointed out that the identified monitoring requirements are not reflected by existing monitoring approaches. Upon this, a dynamic, flexible and adaptable monitoring approach was suggested here. It is based on an mathematical system model the state space. The state space represents the knowledge about the target system that is monitored during run-time. Furthermore, a cyclic monitoring process chain was suggested that improves and strengthens the knowledge respectively the state space over time. This state space is mapped by means of data mining technologies respectively supervised learning into ECA rule sets. These rule sets are used to classify continuously arriving state vectors as normal, abnormal or anomalous. To achieve a dynamic and flexible monitoring solution we suggested a plug-in based approach.

10 Acknowledgments

We wish to thank and acknowledge DLR, ESA and ASTRIUM Space Transportation for their insights and support, with special thanks to Enrico Noack.

References

1. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and Issues in Data Stream Systems. In: PODS ’02: Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. pp. 1–16. ACM (2002)
2. Bellmann, R.: Adaptive Control Processes. Princeton University Press (1961)
3. Bifet, A., Kirkby, R.: Data Stream Mining - A Practical Approach. Tech. rep., Centre for Open Software Innovation (COSI) - Waikato University (2009), `http://moa.cs.waikato.ac.nz/wp-content/uploads/2010/05/StreamMining.pdf`
4. Bolles, A., Geesen, D., Grawunder, M., Jacobi, J., Nicklas, D., Appelrath, H.J.: Sensordatenverarbeitung mit dem Open Source Datenstrommanagementframework Odysseus. In: GI Jahrestagung (2). pp. 404–409 (2010)
5. Borgelt, C., Klawonn, F., Kruse, R., Nauck, D.: Neuro-Fuzzy-Systeme: Von den Grundlagen knstlicher Neuronaler Netze zur Kopplung mit Fuzzy-Systemen. Vieweg (2003)
6. Boslaugh, S., Watters, P.A.: Statistics in a Nutshell. O’Reilly (2008)
7. Carney, D., Çetintemel, U., Cherniack, M., Convey, C., Lee, S., Seidman, G., Stonebraker, M., Tatbul, N., Zdonik, S.: Monitoring Streams: A New Class of Data Management Applications. In: VLDB ’02: Proceedings of the 28th International Conference on Very Large Data Bases. pp. 215–226. VLDB Endowment (2002)
8. Chakravarthy, S., Jiang, Q.: Stream Data Processing: A Quality of Service Perspective. Springer (2009)
9. Chandola, V., Banerjee, A., Kumar, V.: Anomaly Detection: A Survey. ACM Comput. Surv. 41, 15:1–15:58 (2009)
10. Demers, A.J., Gehrke, J., Panda, B., Riedewald, M., Sharma, V., White, W.M.: Cayuga: A General Purpose Event Monitoring System. In: CIDR. pp. 412–422 (2007)
11. Dittrich, K.R., Gatziu, S., Geppert, A.: The Active Database Management System Manifesto: A Rulebase of ADBMS Features. SIGMOD Rec. 25(3), 40–49 (1996)
12. EsperTech: Esper (2011), `http://www.espertech.com/products/esper.php`, online: 30.03.2011
13. Etzion, O., Niblett, P.: Event Processing in Action. Manning Publications Co. (2010)
14. Golab, L., Özsu, M.T.: Data Stream Management. Morgan & Claypool Publishers (2010)
15. Kargupta, H., Bhargava, R., Liu, K., Powers, M., Blair, P., Bushra, S., Dull, J., Sarkar, K., Klein, M., Vasa, M., Handy, D.: VEDAS: A Mobile and Distributed Data Stream Mining System for Real-Time Vehicle Monitoring. In: Proceedings of the Fourth SIAM International Conference on Data Mining (2004)
16. Marwedel, P.: Eingebettete Systeme. Springer-Verlag (2007)
17. Noack, E., Belau, W., Wohlgemuth, R., Mller, R., Palumberi, S., Parodi, P., Burzagli, F.: Efficiency of the Columbus Failure Management System. In: AIAA 40th International Conference on Environmental Systems (2010)

18. Noack, E., Noack, T., Patel, V., Schmitt, I., Richters, M., Stamminger, J., Sievi, S.: Failure Management for Cost-Effective and Efficient Spacecraft Operation. In: Proceedings of the 2011 NASA/ESA Conference on Adaptive Hardware and Systems. AHS ’11, IEEE Computer Society (2011), to appear
19. Noack, T.: Echtzeitberwachung und Langzeitanalyse mittels eingebetteter Systeme. In: Proceedings of the 23nd GI-Workshop on Foundations of Databases (2011), to appear
20. Peckol, J.K.: Embedded Systems: A Contemporary Design Tool. John Wiley & Sons (2007)
21. Schmitt, I.: QQL: A DB&IR Query Language. The VLDB Journal 17, 39–56 (2008)
22. Tsai, J.J.P., Yang, S.J.H.: Monitoring and Debugging of Distributed Real-Time Systems. IEEE Computer Society Press (1995)
23. Witten, I.H., Frank, E., Hall, M.A.: Data Mining: Practical Machine Learning Tools and Techniques. Elsevier (2011)
24. Wolf, F.: Behavioral Intervals in Embedded Software: Timing and Power Analysis of Embedded Real-Time Software Processes. Kluwer Academic Publishers (2002)