A Generic Synchronized Data Acquisition Solution for Distributed Automation Systems

Conference Paper · September 2012

DOI: 10.1109/ETFA.2012.6489655

5 authors, including:

Florian Pethig
Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IO…

Björn Kroll
Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IO…

Oliver Niggemann
Helmut Schmidt University / University of the Federal Armed Forces Hamburg

Alexander Maier
Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IO…

A Generic Synchronized Data Acquisition Solution for Distributed Automation Systems

Florian Pethig, Björn Kroll and Oliver Niggemann
Fraunhofer IOSB-INA,
Application Center for Industrial Automation, Lemgo, Germany
{florian.pethig,bjoern.kroll,oliver.niggemann}@iosb-ina.fraunhofer.de

Alexander Maier, Tim Tack, Matthias Maag
Ostwestfalen-Lippe University of Applied Sciences, inIT - Institute Industrial IT
alexander.maier@hs-owl.de, {tim.tack, maagma}@stud.hs-owl.de

Abstract

This paper presents a novel approach for data acquisition in distributed, heterogeneous automation systems as a basis for new condition monitoring, anomaly detection and visualization applications. The described solution enables process data acquisition in real time Ethernet systems, using the Precision Time Protocol (PTP) from IEEE 1588 for synchronization and the OPC Unified Architecture for data presentation.

As a proof of concept the architecture is implemented using standard hardware components and a x86/x64 architecture without any special operating system. As verified later this setup already achieves a synchronization accuracy of <10ms, sufficient for the majority of production processes to be monitored.

1 Introduction

Data acquisition is the neglected stepchild of the thriving academic field of plant monitoring and diagnosis. While publications for plant monitoring and diagnosis come by the dozen [27, 11, 18, 9], publications about the essential first step for monitoring and diagnosis are hard to find: Without a data acquisition solution for distributed production plants, monitoring and diagnosis algorithms are practically pointless.

Most monitoring and diagnosis projects use either artificial data or data which has been acquired via proprietary acquisition strategies such as additional UDP blocks in plant controllers or via rather old-fashioned protocols such as OPC [1]. But these strategies do not scale for industrial every day production processes since they require high implementation efforts. Furthermore, such solutions have several inherent problems:

(i) Heterogeneous and Distributed Automation Systems: Any system-wide data acquisition solution faces the heterogeneity and distribution of current automation systems: Different plant modules use different control devices, different automation network protocols, and offer different information to the user.

(ii) Non-Synchronicity: Sensors, actuators, and controllers in distributed automation systems do not have a common time base. In most cases, only selective data is available from some parts of the automation system. So, current data acquisition solutions can often not identify the overall plant status at a specific point in time, i.e. no system-wide and synchronized view onto the plant is available.

(iii) Proprietary Access to the Data: The access to the acquired data and the data descriptions are often implemented in a proprietary way. E.g. proprietary web interfaces are used. Furthermore, often the data comes without any semantic information, e.g. information about the sensor's characteristic curve or information about the corresponding process. But, without such semantic information, no automatic interpretation and analysis of the data is possible.

In this paper, a novel solution is presented, which addresses all these problems (see also figure 1):

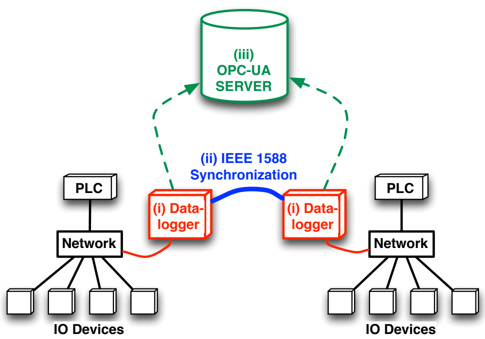

Figure 1. General solution approach.

Solution (i): Sensor and actuator signals are captured by so-called dataloggers. These dataloggers are connected to all relevant segments of the real-time networks and lis-

ten passively, i.e. without influencing the network traffic, to the communication between sensor/actuators and controllers. Furthermore, a multi-layer software architecture allows for a fast adaptation to new network protocols such as PROFINET [23] or EtherCAT [2].
Solution (ii): For a first time, the Precision Time Protocol (PTP) defined in IEEE 1588 [13] is applied to the synchronization of such dataloggers.
Solution (iii): The OPC Unified Architecture (OPC-UA) [7] is used to offer all captured data via a uniform interface. Such an OPC-UA server integrates the data from the different dataloggers, enriches the data with semantic information (using OPC-UA's information model) and makes the data accessible via OPC-UA standardized protocols such as HTTP/SOAP (e.g. for mobile devices) or the OPC-UA binary protocol (e.g. for MES/SCADA systems). Sources for semantic information which can be used include sensors providing semantics themself, MES and SCADA-Systems, IEC 61131 Engineering-Tools as well as manually added information.

This paper is organized as follows: The architecture of this solution is first sketched in section 2 and afterwards compared to the state-of-the-art in section 3. The performance of the solution is analyzed in section 4, while section 5 shows some exemplary monitoring and diagnosis applications, which use the data acquisition solution from this paper. Finally, section 6 concludes this paper by evaluating limitations and advantages, as well as classifying the significance of the presented solution.

2 Architecture

2.1 A Generic Solution Architecture

A generic solution architecture should facilitate data acquisition for every possible plant configuration. Regardless of different fieldbus systems, industrial networks, and configurations the architecture should enable simple acquisition and storage in a defined format. Here, a new architecture for data acquisition in distributed automation systems is presented. As a proof of concept it has been implemented on the basis of existing hardware components and a standard Windows OS. Figure 2 shows the components of the developed architecture: The datalogger software, OPC-UA as a data storage solution, the import of semantics using an AutomationML file and the industrial network to capture data from.

The datalogger implements solutions (i), (ii) and (iii) from section 1 as follows:
(i) Heterogeneous and distributed automation systems are addressed by the Network Abstraction Layer, implementing different industrial network protocols, e.g. Modbus/TCP, PROFINET and EtherCAT.
(ii) Non-Synchronicity is adressed by realizing an IEEE 1588 based synchronization between different dataloggers. This is described in detail in section 2.2.
(iii) Proprietary data access is avoided by using a semantic layer able to import semantics, e.g. using AutomationML.

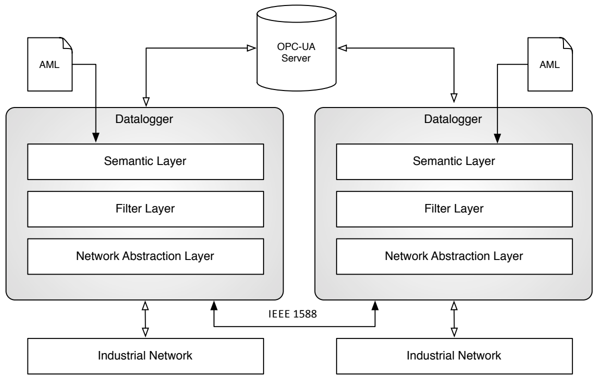

Figure 2. Datalogger software architecture.

Semantics can additionally be applied to the OPC UA information model defined in figure 6, which enables storing of semantically enriched process data, accessible via a uniform interface (OPC UA).

2.2 Synchronized Distributed Measurements in Heterogeneous Automation Networks

Figure 3 shows an example for a typical approach used in industrial manufacturing: A shop floor consists of different production cells, which again consist of different production modules (PM). Typically, a highly heterogeneous structure of industrial network technologies, i.e. fieldbus and industrial Ethernet systems, exists in such environments. Machinery from different manufacturers uses different industrial networks and often also the interprocess communication between production cells is realized using different protocols.

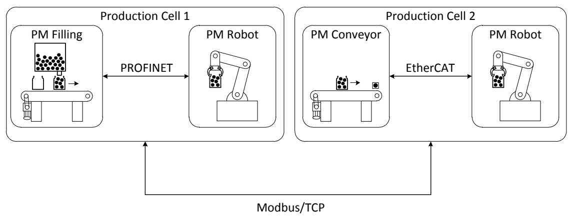

Figure 3. Shop floor communication.

Acquiring process data in such heterogeneous environments is supported by the modular structure of the presented datalogger architecture. Many probes can provide their data to one OPC UA data storage, making it possible to capture traffic in various network segments using different protocols. To establish a uniform time base these probes need to be synchronized.

Although the developed software is generally suitable for every industrial network protocol, we first focus on protocols based on Real Time Ethernet (RTE). Using existing hardware components, it is possible to capture RTE traffic with nanosecond accuracy [12], supported by a hardware timestamping unit generating timestamps on layer 2 of the ISO/OSI reference model. Additionally for distributed networks the PTP can be used for synchronization, enabling an accuracy in the range $< 1\mu s$ [13]. The complete architecture is shown in figure 4, in which state-of-the-art RTE hardware is combined in a x86/x64 PC. The netAnalyzer card provides a network tap for passive capturing without influencing the monitored network. This approach supports that network configurations remain unchanged and no control software has to be adapted. It also includes the mentioned hardware timestamping unit for generating highly accurate timestamps. An Application Programming Interface (API) for the netAnalyzer allows the processing of captured frames and is used by the developed datalogger software component in order to gain access to the network traffic. To add the time synchronization via IEEE1588/PTP, a software stack for the Windows OS can be used to synchronize several datalogger PCs with the needed accuracy. The stack enables the synchronization of distributed network interface cards (NICs) based on the Intel 82574L chipset and can also be accessed via an API. Using the components as described, dataloggers distributed over a network should be able to capture Ethernet frames using accurate and synchronized timestamps.

But one questions remains: At which bit position can the process variables be found in the frames?

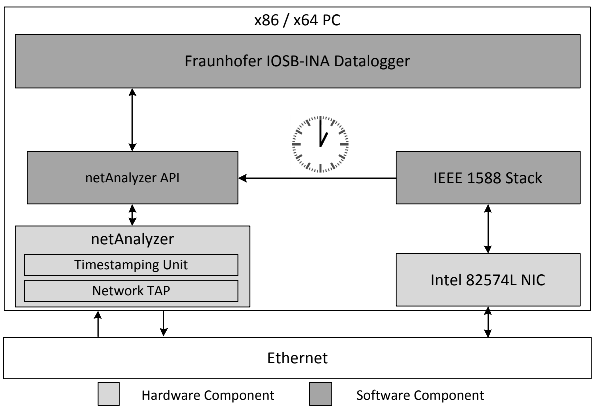

Figure 4. Datalogger components.

Capturing Ethernet traffic is not equal to process data acquisition. To extract process variables from Ethernet frames, the bit position in the appropriate frames must be known, i.e. the protocol and the actual network configuration. Furthermore, all variable names should be extracted from engineering tools and additional available sources for semantic information should be used.

As a proof of concept this approach has been realized for the industrial network protocol PROFINET and the engineering tool PC WORX. This implementation is shown in figure 5 and consists of an offline and online part which are described separately.

Offline integration: The bus configuration of a PROFINET (PN) network and the process variables used in the IEC 61131-3 control program are imported into the datalogger software. In order to import the variables, the following three steps are required: (1) The bus configuration has to be exported out of PC WORX. (2) As a result a CSV-file, which includes variable names and the connection types between PROFINET devices and I/O modules, has to be parsed. The bit positions (offsets) and data types are used to (3) determine the position and length of variables in specific PROFINET IO data objects.

Online integration: As PROFINET handles IO data by exchanging IODataObjects between PROFINET devices, these objects will be defined automatically by PROFINET. The definition of these objects and how they are transferred in the PROFINET Ethernet frames has to be analyzed during the start-up phase of the PROFINET system. This procedure enables acquiring process data in PROFINET networks, which in a next step is stored using an OPC UA server. Using OPC UA as a data storage solution implies several advantages, which section 2.3 describes in detail.

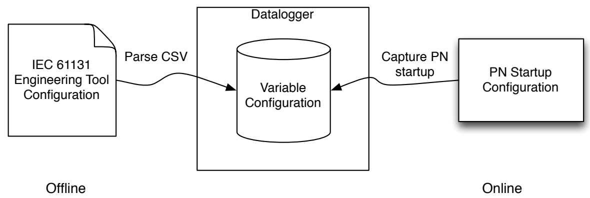

Figure 5. Datalogger off-/online separation.

2.3 OPC-UA as Uniform Data Server

Data inside of an OPC UA server is represented by an explicit information model. A defined semantic is used, realized by a set of metamodels for data acquisition and storage [20, 22]. Using this defined semantic not only acquired measurement values (e.g. current or voltage levels), but also their meanings in the real world can be stored. E.g. additionally to a certain sensor name, value or timestamp the sensor's geographic location, type, temperature, currently active processes, products to be produced, related measurements etc. can be described. The information model used in the OPC UA server component of the datalogger architecture (figure 2) is shown in figure 6. It extends the original OPC UA model with features, such as automatic monitoring on value request. If values are requested, it is possible to choose from monitored or historic values, depending on the application and the required data quality. In addition, values, identifiers and display names were adapted from the OPC UA model and reformatted to match into the environment. These adapted values are shown in figure 6 as requirements. The Unified Modeling Language (UML) notation allows a quick integration of such extensions into the OPC UA framework, since they can be compiled directly as an OPC UA class.

The information model not only implies a self-description-ability, but it is also reflectable at runtime. This means that nodes in such models can change their own structure, which makes it possible to add new devices, structures and/or variables during runtime. A use-

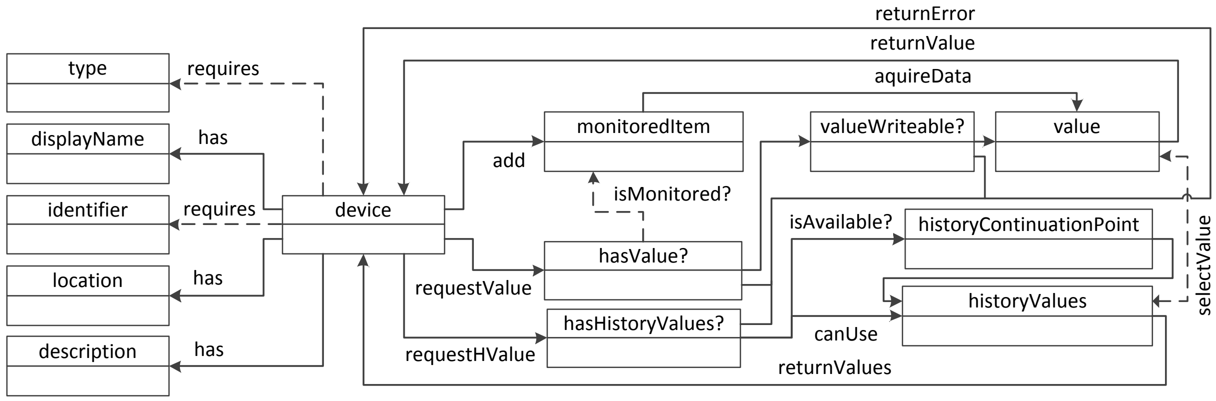

Figure 6. Datalogger OPC UA datamodel.

case describing the significance of reflectable information models for data acquisition in industrial networks could be following dynamic changes in a plant's configuration. Sensors, actuators or even whole production modules will be added and removed dynamically in the future. And because the information model is able to adapt such changes, a consistent data acquisition can still be guaranteed. Advanced data acquisition based on information models therefore enables automatic analysis of plant data.

3 State of the Art

3.1 Data Acquisition

Process data acquisition in distributed systems can generally be done in two different ways: (I) An additional measuring system, including sensors for all important process parameters, can be installed or (II) the existing automation system can be instrumentalized.

Here, method (II) is preferred because often sensors exist which are used in the automation system and could also be used for process data acquisition. The main challenge regarding this approach is the heterogeneous nature of today's automation systems (compare section 1) [6].

Many incompatible technologies are used, e.g. bus systems, industrial Ethernet and time synchronization standards. To overcome this issue, process data acquisition is often done by technologies such as OPC [3] or IEC 61131-3 function blocks, e.g. for writing in a SQL (Structured Query Language) database or to send process data to a SCADA- (Supervisory, Control And Data Acquisition) or History-Server. These approaches add interoperability between components of a heterogeneous automation system on application level, which is why high-accuracy process data acquisition cannot be achieved using these technologies [5].

An alternative is acquiring process data on the field level, which means capturing (Ethernet) frames. For this purpose, several devices exist, which allow frame capturing in a very accurate way [12]. Using these technologies, process data is acquired indirectly, since it still has to be extracted out of the frames. An additional advantage of this approach is that the configuration of the automation system, e.g. the control software, does not have to be changed. When adding a process data acquisition system to a plant retrospectively, having to change a configuration is often the reason for choosing method (I) over (II).

Also for some industrial Ethernet protocols, special profiles exist for energy data acquisition, which is one subset of process data acquisition. These profiles define a consistent format for energy data acquisition on the field level and additionally already provide energy management functions, like switching off machinery in a controlled manner. If such energy profiles are used, consumers (e.g. electric drives) can provide data to an energy management system themself. The consumers are themself peers in an industrial network and no I/O modules are needed anymore. These profiles so far exist for two industrial Ethernet solutions: PROFINET and SERCOS III [25, 26].

For accurate time synchronization in industrial networks, the *Precision Time Protocol (PTP)* defined in IEEE 1588 can be used [13]. Well established industrial Ethernet protocols, e.g. EtherCAT or PROFINET, use their own solutions to achieve synchronization. PROFINET uses a special profile of the *PTP* enhanced for line topologies: The *Precision Transparent Clock Protocol (PTCP)*. EtherCAT also uses a modified *PTP*: The distributed clocks methodology [2].

Interoperability between automation systems will be a challenge for quite a while. Up to now, methodologies for a time consistent process and energy data acquisition in heterogeneous automation systems are missing.

3.2 Information Modeling and Semantics

In the field of information technology, semantic data models are a prerequisite for describing data from various sources. Examples for modeling languages are the UML and the Systems Modeling Language (SysML) [4, 28]. These languages help to describe reality by means of an explicit model, enabling automatic data analysis and interpretation.

For automation technology several extensions based upon the mentioned languages exist. Some of them focus on the design and engineering phase of automation systems (AutomationML [8], Collada [14], PLCopen [24]), while others are only available during runtime (e.g. OPC UA [7], Common Information Model (CIM) [17]). As Legat et al. have shown, especially OPC UA is well suited for information modeling, e.g. for sensor recognition and data provisioning, in industrial automation [15].

4 Results

To evaluate the results, achievable by using state-of-the-art hardware and a standard Windows OS, tests regarding synchronization accuracy and data storage capabilities were carried out.

4.1 Data Synchronization Accuracy

To test the accuracy of the realized data synchronization, the datalogger architecture described in section 2 and depicted in figure 4 running on two Windows 7 PCs (x64) has been used. Frames were generated with a frequency of 1 Hz by using the frame generator MD1230B (Anritsu) and captured simultaneously by both datalogger devices. Here, the network TAPs of the dataloggers were connected by using 2m patch cable. As described in section 2, the IEEE 1588 synchronization is based on the 82574L network interface card (NIC) chipset and a special software stack for the Windows OS. The 82574L chipset supports hardware timestamping, which is a prerequisite for IEEE 1588 synchronization. The datalogger software stack reads the synchronized timestamp from the 82574L NIC and forwards it to the netAnalyzer as initial time for the data acquisition. As a first proof of concept regarding the general architecture, no real time extension for the Windows OS is used. Consequently reading the timestamp out of the 82574L NIC and forwarding it to the netAnalyzer is a non-deterministic process, influencing the achievable synchronization accuracy.

In detail N = 31086 frames were sent and the difference between both datalogger-timestamps was analyzed. The results are shown in figure 7. The arithmetic mean of all timestamp-differences was 2.3 ms with a standard deviation of 1.2 ms.

As a result, a data synchronization accuracy of < 10ms can be achieved by using standard hardware components and a standard Windows OS. This is sufficient for the majority of factory automation processes to be analyzed today, except motion control applications.

4.2 Data Storage Options

Process data acquisition not only means to capture data exchanged between PLCs and IO devices, but also to store this data for later analysis. The integration of part 11 (Historical Access) [21] of the OPC UA specification does not define a data storage solution to be used. Nevertheless, it requires an overall performant data storage for

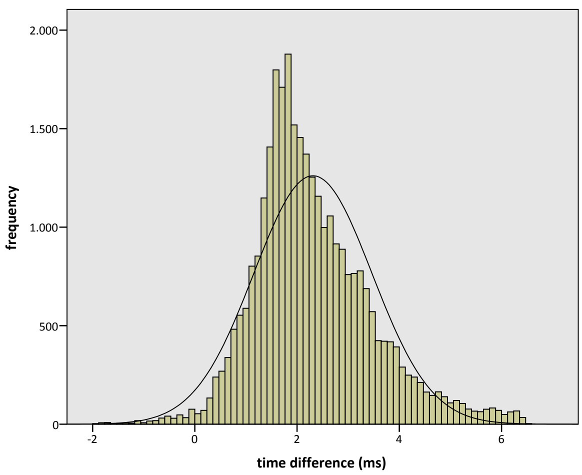

Figure 7. Data synchronization accuracy.

read/write operations. At a first glance, writing single values combined with random data access is a key feature of key/value stores like db4o, redis or leveldb. To confirm this assumption, benchmarks including a key/value store and classic relational database management systems were carried out.

To evaluate the data storages, db4o, SQL Express, SQL Compact, and MySQL time for reading and writing values was taken using the system time of the Windows OS. The measurements are shown in figure 8 and 9.

Compared to each other, classic databases have a overall larger access time but deliver a more balanced read/write behavior. Key/value stores provide the best write performance as shown in figure 9, but lack in read performance as shown in figure 8. For data acquisition purposes, the most important factor regarding data storages is the write performance, for which key/value stores seem to be most qualified.

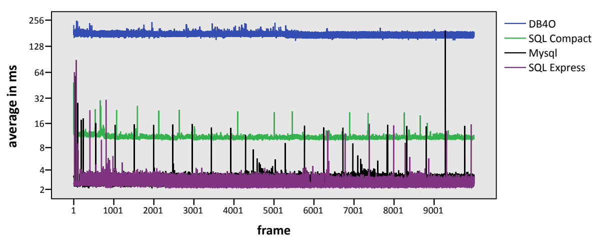

Figure 8. Read performance comparison.

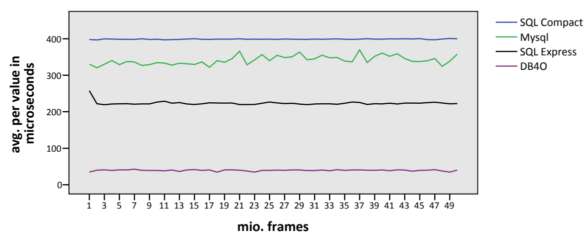

Figure 9. Write performance comparison.

5 Application Examples

In this section application examples for the described datalogger solution are presented, which are intended to point out the significance of synchronized data acquisition and uniform data interfaces in distributed automation systems. These examples range from visualization to complex condition monitoring purposes.

5.1 New Human-Machine Interfaces

Monitoring production plants becomes more and more complex. E.g. an operator needs to observe hundreds of changing values on a variety of screens to monitor a modern production process.

In order to support the process monitoring, providing the operator with an abstract view on the process is a first step. For this, visualization techniques can be utilized. Figure 10 shows an example process visualization, which is created with the help of the principal component analysis (PCA). This visualization concept is introduced in [16]. Each point in the visualization represents the unitless systemwide signal vector at a certain point in time. Observing the visualization, the operator is able to recognize the three main phases "Standby", "Cooldown" and "Production" of this process in an abstract way.

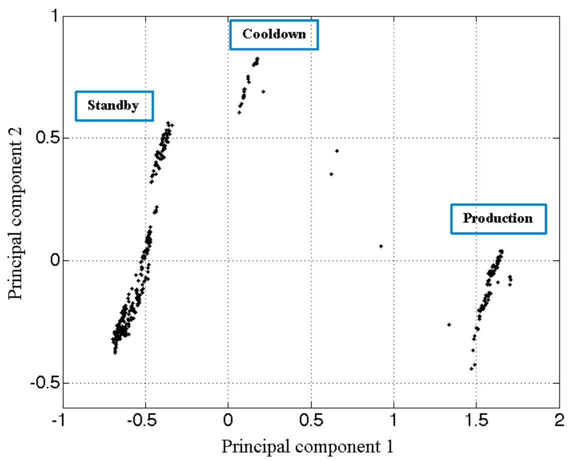

Figure 10. Reference process.

Once the visualization is created, it could be used for condition monitoring, e.g. detecting anomalies. In figure 11 the visualization technique is applied to a process, that comprises an anomaly. The signal vectors which cause the anomaly form a new cluster, thus the operator is able to detect the anomaly by observing the visualization. To enrich the abstract visualization, signals with irregular values could be displayed. An example for this can be seen in figure 11: The signal which leads to the anomaly, a current peak, is depicted with respect to the process timeline. This allows the operator to monitor the entire process on different layers of abstraction and retrieve a more detailed knowledge. With its help, the operator is able to trace the anomaly's origin.

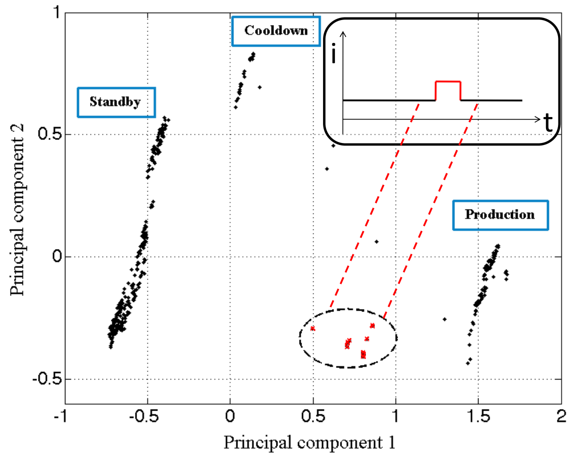

Figure 11. Process comprising an anomaly.

The algorithms, used to create such visualizations, analyze systemwide process data at a certain point in time. Therefore, a consistent process visualization is possible, only if all data sources are synchronized. Considering a modern process setting, this is typically not the case, since a process is formed by heterogeneous interconnected modules. The datalogger presented in this paper provides a time synchronized datasource. With its help, the algorithms used for visualization are able to create a consistent process overview based on systemwide data.

5.2 Anomaly Detection

Generally, anomalies can be detected by means of a complex system model, learned by applying e.g. (hybrid) timed automata [19, 29]. It represents the normal behavior of a system and makes it possible to detect anomalies. Learning such a model is organized in three steps (see also figure 12):

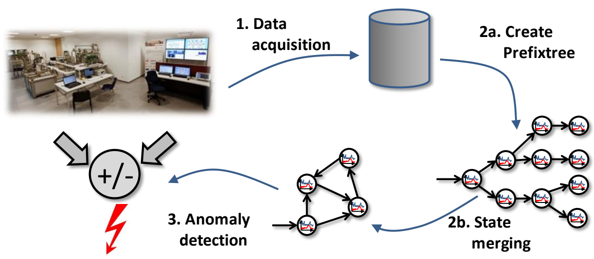

Figure 12. Anomaly detection in three steps.

1. Data acquisition: System-wide data has to be acquired.

2. Model learning: The model learning procedure is divided into two steps: *(a)* The acquired data is stored using a prefix tree in which each event creates a new transition and a new state. Common prefixes are stored only once. *(b)* In the state merging phase pairs of states are checked for compatibility. In the case of compatible states, they are merged. The final automaton represents the normal behavior of the system, e.g. a production plant.

3. Anomaly detection: The learned model is then used to predict the normal behavior while the real behavior is observed using the same data acquisition solution applied in step 1. Every deviation between the real and predicted behavior is signaled as an anomaly, which is why an exact data acquisition is key for a reliable anomaly detection [10].

In detail, model learning and anomaly detection put the following requirements to a suitable data acquisition solution:

Synchronization: To create an overall model, all data from different modules should be combinable. Therefore, a synchronization with an accuracy sufficient for the respective process is required.

Accuracy: To obtain a precise model of the normal behavior and to perform a successful anomaly detection, an exact data acquisition is necessary. Therefore, timestamps need to be generated with an accuracy sufficient for the respective process.

Abstraction: Model learning and anomaly detection algorithms [10] should be able to access plant data consistently via a uniform interface. Especially, an explicit information model supports to adapt to dynamic configuration changes in a plant's automation system.

5.3 Mobile Devices

Using OPC UA as a uniform data interface also enables new applications like visualization of complex production processes on mobile devices, e.g. smart phones. For demonstration purposes, clients for the Android and iOS OS have been developed, implementing not the whole but required parts of the OPC UA specification. These implementations are based on the HTTP/SOAP protocol supported by the OPC UA. Clients can subscribe to variables and visualize values in textual form, as well as in diagrams (see also figure 13). Future applications for process data access via mobile devices could include:

- **Maintenance:** Technicians can access all necessary data at a glance anytime from anywhere.

- **Location based visualization:** A factory manager can access real-time data for process cells/modules based on his current geographical location in the plant (e.g. using a Wifi positioning system or QR-Tags).

- **Enterprise Resource Planing:** Management decisions can be supported by mobile data access from anywhere.

6 Conclusion

This paper presents a novel approach for data acquisition in distributed automation systems, e.g. for condition

Figure 13. Data Access via Android/iOs.

monitoring and complex visualization applications. Using the described generic architecture, data can be acquired with a synchronization accuracy sufficient for most production processes today (<10ms). Furthermore, the applied OPC UA enables adaption to dynamic configuration changes in monitored automation systems, as well as new applications for visualization, anomaly detection and mobile data access. Future work will assess options for increasing synchronization accuracy. This can be achieved, e.g. by using a real time OS supporting deterministic function calls, in order to take full advantage of the PTP.

References

[1] OPC Data Access Custom Interface Standard Version 3.00, 2003. Data Access Custom Interface Standard Version 3.00.

[2] Industrial communication networks - Fieldbus specifications - Part 5-10: Application layer service definition - Type 10 elements (IEC 61158-5-10:2007), 2008. DIN EN 61158-5-10:2008-09.

[3] OPC Unified Architecture, 2009. DIN EN 62541.

[4] I. Alonso, M. Fuente, and J. Brugos. Using sysml to describe a new methodology for semiautomatic software generation from inferred behavioral and data models. In *Systems, 2009. ICONS '09. Fourth International Conference on*, pages 210 –215, march 2009.

[5] S. Cavalieri and G. Cutuli. Performance evaluation of opc ua. Technical report, University of Catania, Faculty of Engineering, Department of Computer Science and Telecommunications Engineering, 2010.

[6] K. Charatsis, A. Kalogeras, M. Georgoudakis, and G. Papadopoulos. Integration of semantic web services and ontologies into the industrial and building automation layer. In *EUROCON, 2007. The International Conference on "Computer as a Tool"*, pages 478 –483, sept. 2007.

[7] M. Damm, S. Leitner, and W. Mahnke. *OPC Unified Architecture*. Springer-Verlag Berlin Heidelberg, 2009.

[8] R. Drath, D. Weidemann, S. Lips, L. Hundt, A. Lüder, and M. Schleipen. *Datenaustausch in der Anlagenplanung mit AutomationML*. Springer, 2010.

[9] D. Dvorak. Process monitoring and diagnosis. *IEEE Expert*, 1991.

[10] S. Faltinski, H. Flatt, F. Pethig, B. Kroll, A. Vodenčarević, A. Maier, and O. Niggemann. Detecting anomalous energy consumptions in distributed manufacturing systems (in press). In *IEEE 10th International Conference on Industrial Informatics INDIN, Beijing, China*, 2012.

[11] C. W. Frey. Diagnosis and monitoring of complex industrial processes based on self-organizing maps and watershed transformations. In *IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, 2008.

[12] Hilscher. netanalyzer. Website: www.de.hilscher.com, 2011.

[13] IEC/IEEE. Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEC 61588 First edition 2004-09; IEEE 1588*, 2004.

[14] Khronos Group. COLLADA - 3D Asset Exchange Schema, February 2011. www.khronos.org/collada/.

[15] C. Legat, C. Seitz, and B. Vogel-Heuser. Unified sensor data provisioning with semantic technologies. In *IEEE 16th Conference on Emerging Technologies & Factory Automation (ETFA)*, 2011.

[16] A. Maier, T. Tack, and O. Niggemann. Visual anomaly detection in production plants. In *(To be published) 9th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*. Rome, Italy, Jul 2012.

[17] A. McMorran. An Introduction to IEC 61970-301 & 61968-11: The Common Information Model. Technical report, Institute for Energy and Environment, Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, UK, 2007.

[18] S. Narasimhan and G. Biswas. Model-based diagnosis of hybrid systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 37(3):348 – 361, May 2007.

[19] O. Niggemann, B. Stein, A. Vodenčarević, A. Maier, and H. 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.

[20] OPC Foundation. OPC Unified Architectur Part 1: Overview and Concepts Release 1.01, 2009.

[21] OPC Foundation. OPC Unified Architectur Part 11: Historical Access Release 1.01, 2009.

[22] OPC Foundation. OPC Unified Architectur Part 3: Address Space Model Release 1.01, 2009.

[23] R. Pigan and M. Metter. *Automating with PROFINET: Industrial Communication Based on Industrial Ethernet*. John Wiley & Sons, 2008.

[24] PLCopen, 2011. http://www.plcopen.org/.

[25] PROFIBUS Nutzerorganisation e.V. The PROFIenergy Profile. *White Paper*, 2010.

[26] sercos international e.V. sercos ENERGY. *White Paper*, 2012.

[27] P. Struss and B. Ertl. Diagnosis of bottling plants - first success and challenges. In *20th International Workshop on Principles of Diagnosis, Stockholm*, Stockholm, Sweden, 2009.

[28] A. Tanasescu, O. Boussaid, and F. Bentayeb. Preparing complex data for warehousing. In *Computer Systems and Applications, 2005. The 3rd ACS/IEEE International Conference on*, page 30, 2005.

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