Research Article

Semantic-Oriented Approach to Productivity Monitoring

Krzysztof Sacha[1] and Dariusz Dąbrowski[2]

[1] *Institute of Control and Computation Engineering, Warsaw University of Technology, 00-665 Warszawa, Poland*
[2] *ILABO Ltd., 05-500 Piaseczno, Poland*

Correspondence should be addressed to Krzysztof Sacha; k.sacha@elka.pw.edu.pl

The study focuses on a design of a novel manufacturing execution system, which acquires sensor data, extracts semantic-oriented information and presents the results to the factory decision maker using mobile devices connected to the Internet. The system is composed of a transponder, located on the plant floor, and a server, which can be located on-site or anywhere in the Internet cloud. The authorized user can connect to the server using wired or wireless connection and receive the requested information on-line, through an arbitrary communication device, such as a laptop, a tablet or a mobile phone. The main goal of the system is to provide the manufacturing decision maker or the plant owner with aggregated data, in order to describe the effects of the plant operation, to enable product tracking and tracing and to show the causes of loss, such as accidents, damages and stoppages.

1. Introduction

A manufacturing plant consists of machinery and other equipment arranged into production lines, and operating under the control of local closed loop and PLC controllers, which acquire data from the sensors and yield commands to the actuators of machines and devices. The local controllers communicate through a network (a fieldbus) with supervisory controllers, which monitor and coordinate the plant operation. Such a centralized system which is used to monitor and control the entire site from a remote area is usually referred to as Supervisory Control and Data Acquisition (SCADA) system.

The main tasks of a SCADA system are: Acquisition and logging of the process data from the plant sensors, detection of alarms and abnormal conditions, presentation of the process data to human operators and execution of the operator commands. SCADA systems can be used to control large-scale processes that can include multiple sites, and large distances. If this is the case, then private communication lines for the exchange of data between the system elements are used, in order to ensure the required security level.

The scope of data collected by a SCADA system and the way of presentation match the needs of an operator, who controls the functioning of the plant equipment. This is quite far from the viewpoint of the manufacturing decision makers, who are interested in the effects of the plant operation and the causes of loss, such as accidents, damages and stoppages rather than in values of, e.g., alarm codes generated by a particular PLC controller. Therefore, the plant control systems can be interfaced to a Manufacturing Execution System (MES), which provides the factory decision maker with a semantically-oriented information describing the current state of the plant. The scope of data acquired by a MES system and the way of presentation match the needs of a manager, who controls the plant efficiency.

The main tasks of a manufacturing execution system are: Gathering and aggregating data that describes economic effects of the plant operation and discovering the reasons why part of the potentially achievable effects are lost. MES systems and SCADA servers are usually connected to each other using intranet, which is an enterprise internal computer network protected from unauthorized external access by means of network gateways and firewalls. OLE for Process Control (OPC) technology can be used to implement a gateway between the control and the enterprise networks.

SCADA and MES systems serve different purposes and support different people within a company. Typically, control systems are located on the plant floor and support on-line activities of the process operators, while MES systems support the manufacturing decision maker at the plant, who must schedule resources, analyze the Overall Equipment Effectiveness (OEE), and do product tracking and tracing from materials through distribution.

There are also other stakeholders in a company, who need to access the plant information, not necessarily from the factory control center. Unfortunately, accessing the plant data is usually possible only on the intra-company network. Information sharing among different people in different areas is difficult, complex, costly, and sometimes impossible. The problem can be solved using Internet technologies, which make it possible to develop a low-cost system for accessing the process data and the manufacturing information over the Internet in real-time. The challenge is to integrate the existing SCADA system with the Internet without compromising security.

This paper describes an approach to acquiring semantic-oriented information about the manufacturing plant and presenting the results using arbitrary, may be mobile, communication devices. System for Monitoring, Analysis and Reporting of Production efficiency (SMARP) is designed to provide the user with aggregated on-line data, which describe the effects of the plant operation and show the causes of loss. Our research goals are the following:

- defining a secure system architecture,
- defining protocols for machine to machine communication,
- defining a method for specification of the system tasks.

A general assumption is that the system shall configure automatically, in such a way that no custom programming is needed.

On the one hand, SMARP can be considered an Internet-based manufacturing execution system. On the other hand, it can be viewed as part of the Internet of Things, composed of hardware, including sensors, actuators and communication devices; middleware for data storage, analytics and Machine-to-Machine (M2M) communication; and presentation tools that can be widely accessed on different platforms. The users of SMARP can connect to the system from virtually anywhere, using personal communication devices, such as a personal computer, a laptop, a tablet or a mobile phone.

2. Related Work

ISA 95:2008 Standard on Enterprise-Control System Integration [1] provides a three-layer taxonomy of control, manufacturing execution and business planning systems used within a manufacturing organization. The main components in the control layer are SCADA systems, which are responsible for monitoring and controlling the production processes maintained by the organization. SCADA systems are usually separated from other systems because of security reasons. SCADA technology is old and it has been described in many places, e.g. [2, 3].

An attempt to build an open interface to control systems is OLE for Process Control (OPC) technology, developed by OPC Foundation. The first version of OPC (1998) was tied to Microsoft Dot Net technology, while the latest version of OPC Unified Architecture (2012) is technology independent. The current suit of OPC specifications can be found in [4]. A discussion and a set of efficiency metrics for the OPC communication protocols can be found in [5].

Internet SCADA (iSCADA) is a newer approach, aimed at the use of public Internet infrastructure to combine traditional SCADA design with the open communication protocols, services and data formats in order to deliver cost-effective SCADA solutions. Several iSCADA systems are offered by commercial vendors. Unfortunately, the description available on proprietary Internet sites of the vendors, e.g. [6, 7], gives no detailed insight into the system architecture.

The architecture of iSCADA systems described in [8, 9] consists of SCADA server, web server, a firewall that separates intranet from Internet, and a set of web clients. SCADA server and web server are located in na intranet, at the enterprise side of the firewall. SCADA server supervises PLC controllers working over a control network and communicates with web server over the intranet. The operator station is a local client in intranet. Other clients are located on the Internet. All the clients are computers or mobile phones using a standard web browser to access web server. What we do not like in this architecture is the possibility to access intranet web server from the Internet. If the web server was hacked, then SCADA server has no protection against the hacker and the security of the entire control system was greatly compromised.

A prototype Internet-based SCADA display system for power industry called SpecNET [10] consists of a server, a set of communication units used to interface SCADA systems to the server, and a set of display clients. The communication units have been specifically designed and coded in Java. The clients use Remote Method Invocation (RMI) or socket library to get access to the server. Security architecture has not been described in the paper. The authors recognize the need for three levels of protection: intranet, extranet and Internet, but do not show the exact placement of system elements within these three zones. They suggest using HTTPS protocol to ensure security at the Internet level.

An idea of a distributed web-based control application is described in [11]. The application is local to a control laboratory, and it uses Internet as a means to convey control data between two computers. The main issue of the experiment is transmission speed. Security is not considered in this application.

Internet of Things (IoT) is an idea of giving everyday objects the ability to connect to a data network [12]. Originally, the term was used in 1999 with respect to the application of Radio Frequency Identification (RFID) sensors in the supply chain management [13]. However, the definition has evolved since that time toward an information network of physical objects, which can become active participants in business processes. IoT is composed of three main components: hardware, made up of sensors and actuators with embedded communication capabilities, middleware, which provides storage and tools for data analytics, and semantic-oriented presentation tools [14, 15]. The contents of IoT is an open research area, which includes RFID and other types of smart sensors, M2M communication [16, 17], data storage and analytics tools, and semantic-oriented presentation. A discussion of the service-oriented client communication can be found in [18].

3. SMARP architecture

SMARP is a system, which receives data from the sensors of a manufacturing plant and presents the data to the Internet clients. The flow of data crosses the company's border, which is protected by a firewall that does not allow incoming Internet connections. For this reason, OPC server can-

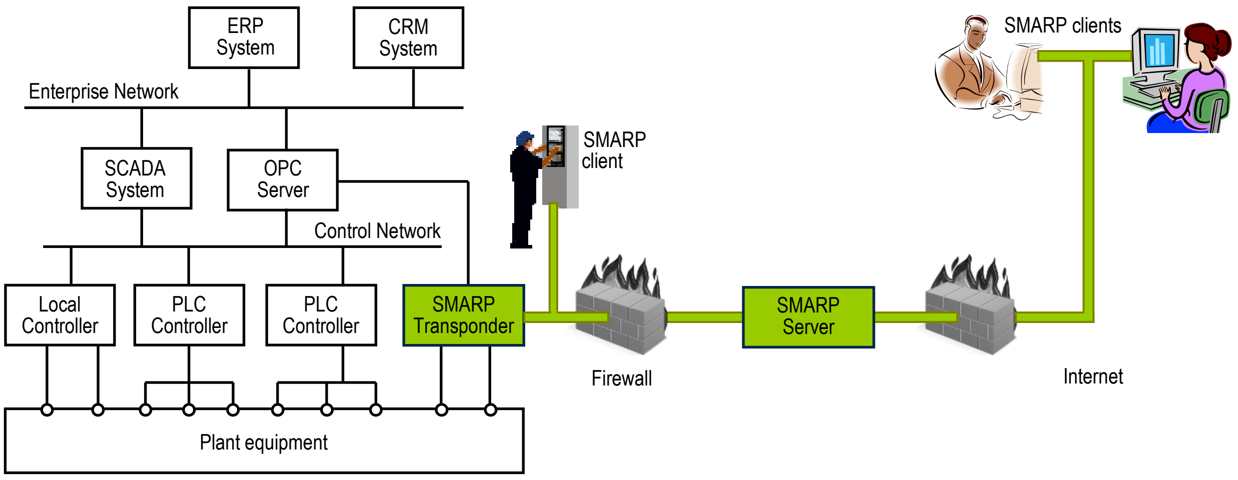

Figure 1. The overall SMARP architecture

not be used as a direct source of the process data harvested by SMARP. Instead, our system consists of two basic units, which reside at opposite sides of the firewall:

- SMARP transponder, which acquires the selected process data items and sends to the server;
- SMARP server, which stores the data, calculates the results and presents to the user.

The transponder resides at the intranet side of the firewall. It maintains a connection to the server and sends the raw process data over this connection. The server resides at the Internet side of the firewall, perhaps in a demilitarized zone. It receives connections from the transponder – to receive the data, and from the users – to present the required information. In fact, more than one transponder can collaborate with the same SMARP server, and more than one plant can be served by a single server.

The transponder is interfaced to the control systems on the plant floor in such a way that it does not interfere with the plant operation. It acts as a sniffer, which acquires data from the controllers as well as from its own sensors, and sends the data to the server over the Internet. The server can be located anywhere in the network, e.g., on the plant floor or in the Internet cloud. The users can connect to the server through a web browser on an arbitrary communication device, such as a laptop, a tablet or a mobile phone.

The actual architecture of SMARP depends on the environment, in which the system is used. Business scenarios can be the following.

A big company which has several production lines and full-fledged control and management systems can maintain a proprietary SMARP server. Figure 1 shows a typical architecture where the server is installed within a demilitarized zone, in order to enhance security of SMARP as well as of other systems. The plant operators, the management staff and other manufacturing decision makers can access the server whether over intranet or over the Internet.

A small enterprise, which maintains a production line controlled by a few local controllers, neither needs nor can afford an expensive SCADA system. If this is the case, then a transponder with only few simple sensors, such as photocells to register the flow of products, can be used to send the process data to the server located somewhere in the Internet cloud. The plant owner can rent the SMARP functionality as a service and pay only small monthly installments instead of buying the system. The service allows the owner or a person authorized by the owner to access the data through a web browser in order to monitor the plant operation from a remote area. SMARP architecture for this business scenario is shown in Figure 2. The firewall can be a dedicated network device, a software program on transponder, or can even be missing.

The communication between SMARP transponder and SMARP server goes through a firewall, which is configured to prevent connections coming from the Internet. Therefore, the originator of the communication is always the transponder, which sets TCP/IP connection up and then uses POST method of HTTP or HTTPS protocols to exchange messages. The data is sent by the transponder in the body of a POST request message. The stream of messages from the transponder to the server is continuous, i.e., the transponder sends messages periodically, with a con-

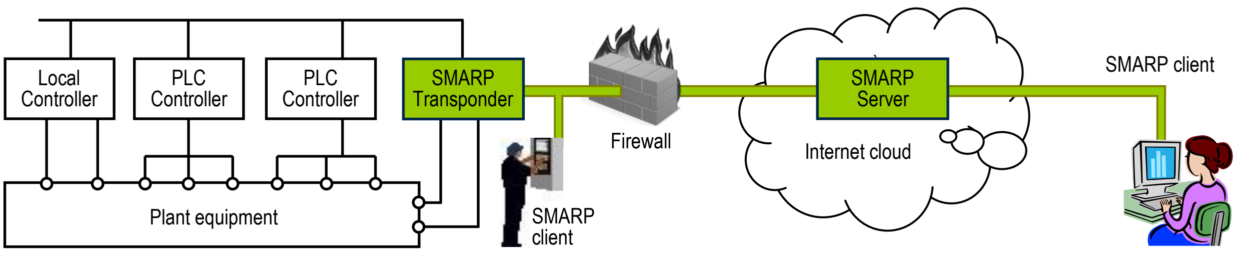

Figure 2. Cloud-based SMARP architecture

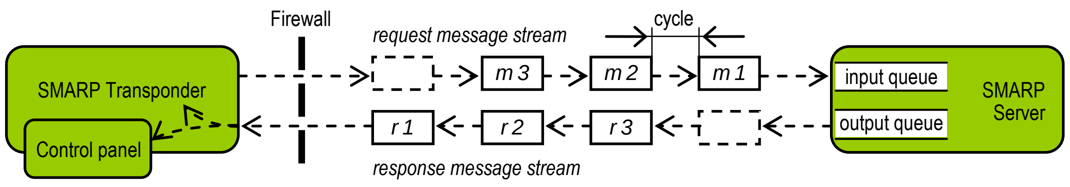

Figure 3. Transponder–server communication

stant period. The content of a message is a sequence of values measured by the sensors attached to machines on the plant floor. If no data is to be sent in a given period, then a heartbeat message is transmitted. The messages are uniquely identified by sequence numbers that enable the server to detect messages that have been lost.

A transponder message is responded by the server according to the request-response message exchange pattern. In most cases, the server will return an empty response message that simply tells the transponder that no errors occurred. Occasionally, the server may use a response message to ask transponder for an action, which can be unrelated to the previous transponder message. SMARP server uses this method to send alarm communicates to a control panel that can be mounted on the plant floor and electrically attached to the transponder. The flow of messages is illustrated in Figure 3. All the messages can be divided into two logical channels, which are used for different purposes:

- the transponder channel consists of transponder messages, which convey the process data, and server messages, which convey the action requests;
- the panel channel consists of messages, which convey communicates targeted to the control panel.

SMARP messages are XML documents inserted into the body of SOAP [19] messages, which are sent using Web services over HTTP/HTTPS protocols. SOAP messages include headers that are used to route the messages through intermediaries. The efficiency of those protocols, as discussed in [5], is satisfactory. The use of Web services over HTTP/HTTPS protocol suite allows SMARP to be mapped to future technologies as necessary, without negating the basic design.

Every SMARP message is a complex XML element, which name identifies the message type. The content of the message is a list of the message data items. The following is an example of a SMARP transponder message.

```
<packet plant = "Mineral Water Ltd."
        from  = "T1"
        date  = "2014-05-16T17:25:02">
  <item name = "T1">
    <value> 12 </value>
  </item>
  <item name = "Pressure"
        date = "2014-05-16T17:22:00">
    <value> 150.5 </value>
  </item>
  <item name = "Count"
        date = "2014-05-16T17:20:00">
    <value> 88 </value>
  </item>
  <item name = "Count"
        date = "2014-05-16T17:22:00">
    <value> 90 </value>
  </item>
  <item name = "Temps"
        date = "2014-05-16T17:20:41">
    <value idx = "0"> 24.3 </value>
    <value idx = "1"> 28.5 </value>
  </item>
  <item name = "Params"
        date = "2014-05-16T17:21:12">
    <value field = "Level"> 2 </value>
    <value field = "Flow"> 7.1 </value>
  </item>
</packet>
```

The transponder message is a complex element called `packet`. The attributes of the element identify the plant where the data were measured (`plant`), the transponder (`from`) and the message sending time (`date`). The latter attribute can be considered as a timestamp of the message. The contents of the element are the measured data samples (`item`). Each data sample defines the name of the process variable (`name`), the measurement time (`date`) and the measured value (`value`). If the `date` attribute of an item is missing, then `date` attribute of `packet` element is used by default. All the names, time stamps and values are encoded in XML using the formats defined in XML Schema specification [20].

The first data item bears the name of the transponder (`T1` in this example). The value of this item is the sequence packet number. The advantage of such a convention is that the message number can be processed by the server exactly as the value of any other item. The next three items are samples of simple variables. Two of those items relate to the same variable (`Count`). This is not an error – the first item is a sample measured at 17:20, and the other one at 17:22. The last two items are samples of complex variables – a one dimensional array and a record.

The advantage of XML-based data encoding is Web service compatibility. The disadvantage is big size of messages, which results in low communication performance. To cope with the problem, transponder messages convey to the server only those values of variables, which have changed since the previous message. The server maintains the current values of variables and updates those values when new message arrives. The full list of variables is transferred only at the start of SMARP system and after a major communication breakdown.

4. SMARP transponder

The transponder acquires data from the sensors mounted at the plant floor, converts the data to the standard formats (data types) and sends the data values to the server. The data are acquired from:

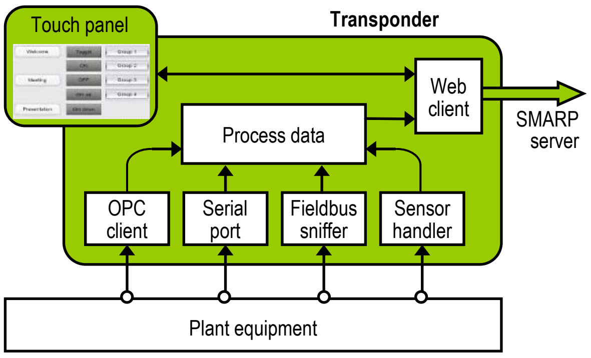

Figure 4. SMARP transponder

- OPC servers, which are part of the SCADA system;
- PLC controllers, interfaced to the transponder by means of serial communication ports;
- control networks, which can be sniffed by the transponder using appropriate network cards;
- SMARP sensors, such as photocells or bar code readers, installed to measure the flow of products.

The acquired data values are stored within the transponder together with time stamps, and then send to the server as described in Section 3. The conceptual model of SMARP transponder is shown in Figure 4.

The transponder can also handle an optional touch screen to communicate with the plant operator. This communication channel is used when the server identifies a stoppage, which cannot be classified on the basis of the sensor data values. If this is the case, then the server asks the operator to classify the type and the cause of the stoppage. The communication is initiated by the server, and the operator acts as an additional "smart sensor".

An event, which initiates the communication, is called an alarm. It is announced to the operator by a server message, which is a complex XML element called `action`. The following is an example of a server message.

```
<action plant = "Mineral Water Ltd."
        to    = "T1"
        panel = "P1">
   <alarm name  = "Alm12"
          date  = "2014-01-17T17:22:00"
          value = "1"
          reply = "Code12"/>
   <note>
      No output in the last 3 minutes
   </note>
</action>
```

The contents of the message define the alarm instance (`alarm`) and the text to display (`note`). The attributes of the `alarm` element give the alarm name, the time of occurrence, the value (1 for alarm activation, 0 for clearance) and the name of the reply variable. The reply of the operator is passed to the server as a value of the reply variable. An example reply message is shown below.

```
<packet plant = "Mineral Water Ltd."
        from  = "T1"
        panel = "P1">
  <item name = "T1">
    <value> 13 </value>
  </item>
  <item name  = "Code12"
        date  = "2014-01-17T17:22:00"
        alarm = "Alm12">
    <value> 3 </value>
  </item>
</packet>
```

The reply message format complies with the transponder message format described in Section 3. However, the message contains two additional attributes: `panel` and `alarm`. The former identifies the source of data and the latter binds the item value to the alarm name. The timestamp of the item equals the timestamp of the alarm occurrence. This enables possibility to bind the reply item value to the particular alarm instance.

5 SMARP server

The server receives a continuous stream of messages that convey process data samples sent by transponders attached to the equipment at the plant floor. The messages are transmitted concurrently in an asynchronous way. To prevent message loss while processing data, the server temporarily stores the incoming messages in an input queue, from which they are then fetched and processed by data and alarm processor module. The conceptual model of SMARP server is shown in Figure 5.

5.1. The architecture of SMARP Server. The current state of the plant is stored in memory as a list of values of the process variables. An item of this list consists of the name

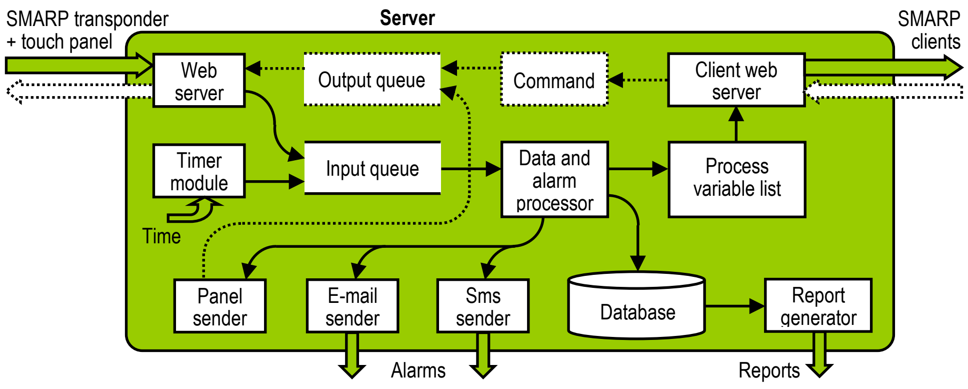

Figure 5. SMARP server

of a variable, the current value and the timestamp. Data and alarm processor looks at each transponder message, reads the samples contained in the message and stores the values and timestamps of the measured variables into the process variable list. In addition, the history of changes is archived in a history log in the server database.

The values of some of the process variables cannot be measured directly, but can be derived from values of other variables. Data and alarm processor computes those values using other variables as arguments. The computation is executed each time when the arguments change. The computed values of the derived variables are processed, stored and archived in the same way as the values of the measured variables.

The processing of data values includes detection of alarm conditions, which are indicated by abnormal values of certain measured or derived variables. If this is the case, then data and alarm processor rises an alarm and sets a binary variable, called the report variable of the alarm, to 1. The value of the report variable and the values of all of its arguments are archived in the history log, together with the name of the alarm and the time of its occurrence. The rise as well as the clearance of an alarm can be communicated to the users via emails, sms or panel messages.

The computation of selected derived variables can also be triggered by a timer module. The module counts time signals with the resolution of 1 s and sets a binary timer variable to 1 when the predefined time interval expires; then, the module places a message with the value of the timer variable into the input queue. The message is processed in the same way as a transponder message.

Client web server accepts a small set of commands that allow the users to control the transponder configuration. For example, an authorized user may change the sampling rate of a measured variable. A user command is packed into an XML element, put into the output queue and sent to the transponder via the web server. An example message of this type is shown below.

```
<action plant = "Mineral Water Ltd."
        to    = "T1">
   <item name   = "Count"
         sample = "5"/>
</action>
```

The command message format complies with the server message format described in Section 4. The contents of this message define the variable (`item`) and the new sampling interval (`sample`).

A single SMARP server can receive messages from several transponders attached to the same or to different plants. The process variables related to the same plant are stored and processed jointly. The process variables related to different plants are handled separately. One can imagine that each plant has a separate instance of the server, which works independently and concurrently with the instances of other plants.

5.2. Semantic information processing. Raw data values received from transponders describe the plant at a very detailed level. The server aggregates the data in order to derive information, which is meaningful to the manufacturing decision maker. The basic unit of computation in the server (a "thing" of IoT) is a manufacturing machine, e.g., a bottle filling machine, a labeler, a packaging machine, etc. A manufacturing machine is described by a set of four attributes:

- *state* – an encoded description of the current state of the machine, e.g., '*working*', '*planned stoppage*', '*unplanned stoppage*', '*failed*', etc.;
- *total* – the total output of the machine, e.g. the number of bottles filled by a filling machine;
- *trash* – the total number of defected output units, e.g., the number of bottles not filled properly;
- *nominal* – the nominal output of the machine per hour.

All the attributes are process variables reported by a transponder (*nominal* can be a constant). The historical values of those variables are archived in a server database. Those values can serve as a basis for automatic generation of reports used to describe the plant operation from the production perspective. Examples of such reports are: OEE within a given period of time, the total quantity of output within a given period of time, the causes of failure, the total stoppage time, the unplanned stoppage time, the short stoppage time, etc. [21]. The reports, as well as trend logs, can be presented to the user on-line or on-demand.

For example, assume that a report of OEE in the period from t_1 to t_2 is required. If this is the case, then the server:

- Retrieves the values of *state* between t_1 and t_2, and calculates the *operating time* as the total time during which *state* = '*working*'.
- Calculates the *planned stoppage time* as the total time during which *state* = '*planned stoppage*'.
- Retrieves the values of *total* at t_1 and t_2 ($total_1$, $total_2$, respectively), and calculates the output:
  $output = total_2 - total_1$
- Retrieves the values of *trash* at t_1 and t_2 ($trash_1$, $trash_2$, respectively), and calculates the spoilage:
  $spoilage = trash_2 - trash_1$
- Calculates the scheduled time:
  $scheduled\ time = t_2 - t_1 - planned\ stoppage\ time$
- Computes the OEE coefficients:
  $availability = operating\ time\ /\ scheduled\ time$
  $performance = output\ /\ nominal\ /\ operating\ time$
  $quality = (output - spoilage)\ /\ output$
  $\mathrm{OEE} = availability \times performance \times quality$

The arrangement of manufacturing machines at the plant floor is modeled within the server as a graph, which nodes are machines and edges are flows between those machines. The graph is described in an XML document. The machines are characterized using attributes explained above, and the connections between the machines are described in GraphML language [22]. Such an XML specification allows automatic creation of synoptic diagrams, which show the plant structure and the current state of the

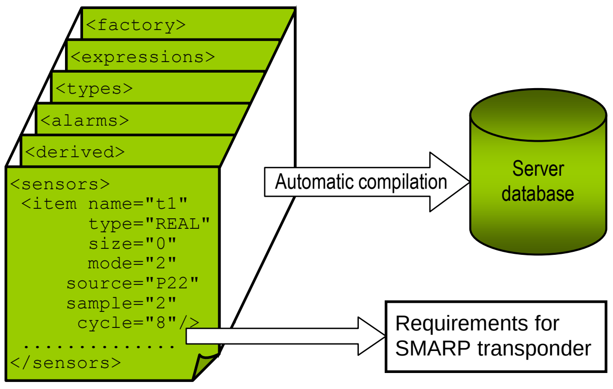

Figure 6. SMARP system specification

plant equipment retrieved from the process variable list (Figure 5). The synoptic diagrams and the results of OEE calculation can be accessed by the end users through an Internet browser, using arbitrary communication devices.

6 Deployment and operation

An instance of a SMARP server consists of a set of data structures and a set of data processing tasks. All of those elements are defined in a system specification, which can be compiled by an automatic tool and deployed in an execution environment. No custom programming is needed in order to put the server into operation. This helps in keeping development time short and development cost low.

SMARP specification consists of a number of XML documents that define data types, measured variables reported by transponders (sensors), derived variables computed by the server, expressions used for computation, alarms, and the plant structure. The documents are compiled and placed into the tables of the server database. The contents of those tables drive execution of an individual server instance. The use of specification documents is schematically illustrated in Figure 6.

The computation performed by the server is triggered by transponder messages, each of which brings new values of a set of the measured variables. The server receives the values and stores in the process variable list (Figure 5). When the value of a measured variable is changed, the server proceeds similarly to a spreadsheet and calculates expressions, which have the variable in the argument list. The values of those expressions are assigned to derived variables or to alarms.

The specification of measured variables is a document, which defines not only server data but also the interface between the server and a cooperating transponder. A measured variable is characterized by a set of attributes, such as *name*, data *type*, the *size* of a complex variable, the *mode* of data processing and storage, the *source* of the data value, the reporting *cycle* to the server and the sampling rate. The latter attribute (*sample*) is optional and is equal to *cycle*, by default. This combination of values means that each data sample is reported to the server. However, if an alarm occurs, the server may automatically change the sampling rate in order to increase the time resolution of data that describe the current plant state.

A detailed description of the structure of all the XML documents is beyond the scope of this paper.

7 Conclusions and future work

SMARP is a lightweight web-based production data acquisition and monitoring system, which goal is to support the manufacturing decision maker of a plant. The system consists of simple transponders located at the plant floor and a server, which can be located anywhere in the Internet. The communication between those two components crosses the security zone created by a company's firewall and is based on XML messages, exchanged using web services and HTTPS protocol. The main part of SMARP is the server, which acts as a manufacturing execution system and provides aggregated process data that describes the effects of the plant operation and shows the causes of loses, such as accidents, damages and stoppages. The user can connect to the server through an arbitrary communication device, e.g. a laptop, a tablet or a mobile phone.

The scope of data acquired by the system and the rules for processing of the data are defined in an XML-based system specification, which is compiled and deployed on the server by an automatic tool. No custom programming is needed to deploy the server and put into operation.

The advantages of the described SMARP architecture are the following.

- The security of control as well as enterprise systems of the company is not compromised.
- Full functionality of a manufacturing execution system is accessible for the authorized user virtually anywhere.
- The cost of the system is low, because no custom programming is needed.
- The architecture and the machine to machine protocols are open and not tied to a particular vendor.
- The flexibility and scalability of the system is high, due to the use of Internet technologies.

The concept of the web-based production monitoring system has been validated in a real industrial environment. Currently, we are working towards the implementation of tools to support automatic generation and deployment of SMARP.

References

[1] ANSI/ISA 95.00.01-2000, "Enterprise-Control System Integration – Part 1: Models and Terminology", 2008.

[2] D. Bailey and E. Wright, "Practical SCADA for Industry", Elsevier, 2003.

[3] S.A. Boyer, "SCADA Supervisory Control and Data Acquisition", ISA: The Instrumentation, Systems, and Automation Society, 2010.

[4] OPC Foundation, "OPC Unified Architecture Specification", http://www.opcfoundation.org/.

[5] R. Kyusakov, H. Mäkitaavola, J. Delsing, and J. Eliasson, "Efficient XML Interchange in Factory Automation Systems", in *Proceedings of the 37th Annual Conference on IEEE Industrial Electronics Society (IECON 2011)*, pp. 4478-4483, November 2011.

[6] Devices World, "iSCADA", http://www.devicesworld.net/.

[7] Vipond Controls. "iSCADA", http://www.vipondcontrols.ca/iscada/.

[8] E. Ozdemir and M. Karacor, "Mobile phone based SCADA for industrial automation", *ISA Transactions*, vol. 45, no. 1, pp. 67-75, 2006.

[9] D. Wallance, "How to Put SCADA on the Internet", *Control Engineering*, vol. 50, no. 9, pp. 16–21, 2003.

[10] B. Qiu, H.B. Gooi, Y. Liu, and E.K. Chan, "Internet-Based SCADA Display System", *IEEE Computer Applications in Power*, vol. 15, no. 1, pp. 14-19, 2002.

[11] A.E. Kuzucuoglu and G. Erdemir, "Development of a Web-Based Control and Robotic Applications Laboratory for Control Engineering Education", *Information Technology and Control*, vol. 40, no. 4, pp. 352-358, 2011.

[12] N. Gershenfeld, R. Krikorian, and D. Cohen, "The Internet of Things", *Scientific American*, pp. 76-81, October 2004.

[13] K. Ashton, "That 'Internet of Things' Thing", *RFID Journal*, 22 June 2009.

[14] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey, *Computer Networks*, vol. 54, no. 15, pp. 2787-2805, 2010.

[15] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions", *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645-1660, 2013.

[16] J. Höller, V. Tsiatsis, C. Mulligan, S. Karnouskos, S. Avesand, and D. Boyle: *From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligenc*e, Elsevier, 2014.

[17] M. Zorzi, A. Gluhak, S. Lange, and A. Bassi, "From today's intranet of things to a future internet of things: a wireless- and mobility-related view", *IEEE Wireless Communication*, vol. 17, no. 6. pp. 44-51, 2010.

[18] Z. Chen, H. Lin, M. Chen, D. Liu, Y. Bao, and Y. Ding, "A Framework for Sharing and Integrating Remote Sensing and GIS Models Based on Web Service", *The Scientific World Journal*, vol. 2014, pp. 1-13, 2014.

[19] W3C, "SOAP Version 1.2 Part 1: Messaging Framework", http://www.w3.org/TR/soap12-part1/, 2007.

[20] W3C, "XML Schema Part 2: Datatypes", http://www.w3.org/TR/xmlschema-2/, 2004.

[21] I. Alsyouf, "Balanced Scorecard Concept Adapted To Measure Maintenance Performance: A Case Studt", in: A.G. Starr and Raj B.K.N. Rao (Eds.), *Condition Monitoring and Diagnostic Engineering Management*, pp. 227-234, Elsevier, 2001.

[22] U.Brandes, M.Eiglsperger, J.Lerner, "GraphML Primer", http://graphml.graphdrawing.org/.