XIX IMEKO World Congress
Fundamental and Applied Metrology
September 6–11, 2009, Lisbon, Portugal

EVENT-BASED DISTRIBUTED MEASUREMENT SYSTEM FOR PQ MONITORING APPLICATIONS

Fabrizio Ciancetta [1], *Edoardo Fiorucci* [1], *Giovanni Bucci* [1], *Carmine Landi* [2]

[1] Dipartimento di Ingegneria Elettrica e dell'Informazione, University of L'Aquila
Poggio di Roio, 67040, L'Aquila, Italy, {fabrizio.ciancetta, edoardo.fiorucci}@univaq.it
[2] Dipartimento di Ingegneria dell'Informazione, Seconda Università di Napoli
Via Roma 29, 81031, Aversa (CE), Italy, {daniele.gallo,carmine.landi}@unina2.it

Abstract – The problem of locate disturbance sources in power network is a critical task. Several methods based on multi point measurement techniques have been proposed in the last few years proving a good contribution on this filed [1-2]. In this paper an event-based distributed measurement system for power network has been developed. The system adopts a hierarchical approach to monitor the power network: the sensing parts are measurement systems which perform the monitoring of Power Quality (PQ) parameters such as Total Harmonic Distortion (THD) and Root Mean Square (RMS); the management part is a supervisor node which uses an asynchronous symmetric communication protocol to exchange information with the devices connected to the network. When an abnormal behaviour has recorded, the measurement system notifies the supervisor node which begins to download the data from every smart sensor connected to it. To give much more freedom end flexibility to the sensors network, the smart sensors use the Web services technology to export its services and functionalities. In this paper, the main parts of the system are explained in details and some experimental results are reported.

Keywords: smart Web sensor, network sensors, Web service, UDDI, symmetric asynchronous system

1. INTRODUCTION

Several applications require for a distributed measurement system able to measure the same or different parameters at different points in a wide area. In recent years, wide varieties of solutions have been proposed for remote measurement and data transmission. Distributed systems based on smart web sensors represent the best solution to many different measurement problems [3-6]. By adopting these sensors, a client can receive the measure of a particular physical quantity with a browser or an application, developed to receive information from the web server.

From developer side, smart Web sensors present always a closed approach to interact with them, so web services are adopted in order to give a standard approach in developing a Service Oriented Architecture. From network side, Internet is a widely adopted network where any user is uniquely identified with its IP address. So, to implement a distributed measurement system, it is necessary to use a common and open communication protocol to exchange information and a methodology to auto-configure any smart sensor is linked to the network. During the development of a smart Web sensors network, two are the main problems to solve: i) how "discover" the smart web sensor and ii) which interface is "published" to consume the services. Our proposal is to use the Universal Description Discovery and Integration (UDDI) technology.

The original vision for UDDI was as standards and tools that would help companies conduct business with each other in an automated fashion. The searchable central registries provide a publishing and subscribing mechanism to store the agency and service descriptions and to point to detailed technical specifications that define the interfaces to these services. The UDDI registry is a possible solution of the previously introduced measurement problems because it provides a standardized method for publishing and discovering information about web services and because it implements a customization of type's description that can be used to standardize the smart Web sensor interface to the network.

In the proposed distributed measurement system, the sensing parts are smart Web sensors which provide a Web service interface to share the acquired data. The smart Web sensors store a technical model (tModel) document in a UDDI registry in which all the information about the exported service are present. The smart Web sensors reference their sensors interface to other tModel to create a categorization of sensors (taxonomy) and so build sensors network linked with the same reference. At the end a supervisor node has been developed to use the proposed network by means of symmetric asynchronous communication technique. The supervisor node sends asynchronous packets to UDDI registry to find smart Web sensor and sends asynchronous packets to smart Web to consume the published services and to the client connect to the network.

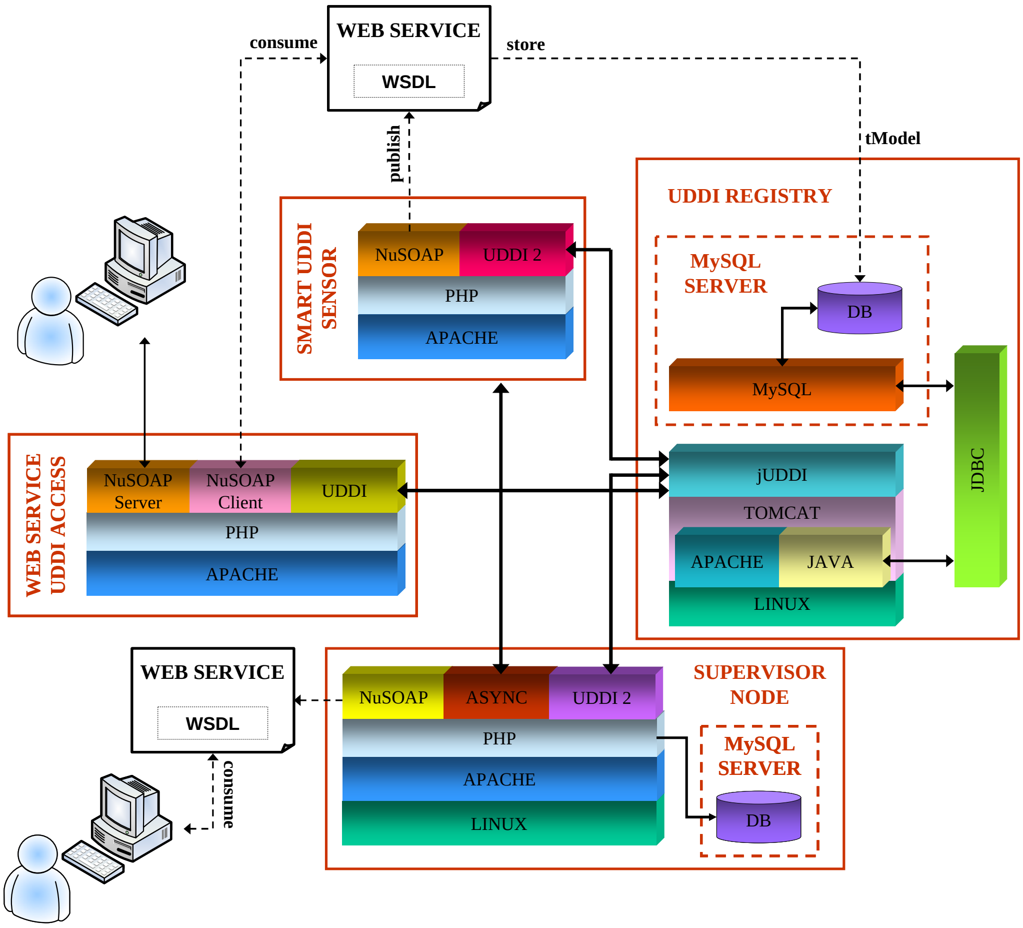

Fig. 1. Network and software architecture of the proposed distributed system

2. NETWORK ARCHITECTURE OF THE PROPOSED DISTRIBUTED MEASUREMENT SYSTEM

The central elements of the proposed network are the UDDI registry, the Web service UDDI access, the smart UDDI sensor and the Supervisor Node.

The UDDI registry is the load-bearing wall of the whole network. It is composed by a personal computer in which a UDDI registry has been implemented.

The smart UDDI sensor is a smart Web sensor which publishes on Internet a Web service of application-depending functionalities and stores in the UDDI registry its functionalities.

The Web service UDDI access is a Web service with three distinct functionalities: i) it allows at a client to access to the network via SOAP messages, ii) it interfaces the client to the UDDI registry to search a particular smart Web sensor and iii) it interfaces the client to the smart UDDI sensor to consume the Web service.

The Supervisor Node is composed by a personal computer in which a Linux based platform has been used to consume the Web services present in the sensors network. A relational diagram of the proposed architecture is reported in the Figure 1, while in the next subsections the main parts of the network elements are explained in more details.

2.1. Taxonomies of Smart UDDI Sensors

In the proposed network, every smart UDDI sensor (SUS) publishes its services both into the Internet network and into the UDDI registry as reported in Figure 1. For the Internet publishing part, the SUS presents a Web service interface allowing every user to access to its functionality as in [3, 7, 8]. For the UDDI registry publishing part, the smart Web sensor stores in a UDDI record all the standard information (such as company name, business identifier, etc...) and technical information (such as tModel) to allow every consumer to search and to find the specific SUS [9].

However, every SUS can provide a different list of services and a different use of the same services caused by a different interface that the smart Web sensor present to the consumer. In the proposed system, a tModel describing the sensor services has been developed as a basic common interface from which every smart Web sensor realizes a concrete implementation [10]. In this way, a taxonomy of smart Web sensors can be developed to create a sensors network [11] that presents the same interface to the client even if they have a different implementation [12].

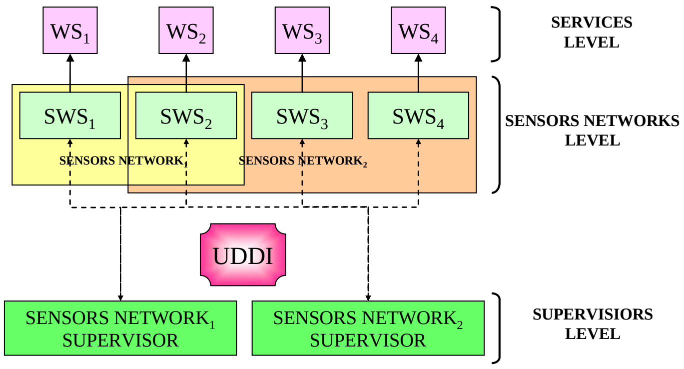

Fig. 2. How supervisors use the sensors network

2.2 Web Service UDDI Access

UDDI adopts a standard protocol based on SOAP to send and receive information [13]. The access to the UDDI registry is defined by the APIs reported in [14]. To give much more freedom to the clients that use the distributed measurement system, a special Web service has been developed to consume the network of SUSs, accessing to the UDDI registry [15]. As reported in Figure 1, the Web service UDDI access (in short SWUA) provides an interface that links the client to the UDDI registry and to the network of SUSs. For the UDDI interface it publishes only the methods that allow the discovering of a SUS, while, for the SUS interface, it publishes some generic method to consume the Web service of SUS involved in the search. Ultimately, the Web service UDDI access provides a complete access to the network of SUSs with some simple tasks: i) the client sends a query to the WSUA using a Web method; ii) the query is forwarded by the WSUA to the UDDI registry using the UDDI APIs; iii) the response of the query is packaged in a software object with many information about the SUSs involved in the search; iv) the client can access to the Web service of SUS utilising an other WSUA's Web method; v) the WSUA sends a SOAP message to the SUS and waits a response that sends back to the client.

2.3 Supervisor Node

The supervisor node structure is reported in Figure 1. It is composed by Linux-based platform on a personal computer over which APACHE e PHP are installed as software services. Three are the main elements that supervisor node uses to communicate with the other parts of the network: i) NuSOAP class is the supervisor node interface which provides all the Web methods to access to the exported data and to access to the sensors network; ii) UDDI2 class is used to communicate with the UDDI registry and iii) the ASYnchronous interface is used to communicate with all the smart Web sensors. By means of symmetric asynchronous communication technique, the supervisor node uses the sensors network as reported in Figure 2: first of all, the supervisor node sends an asynchronous packet to every smart Web sensors (the sensing part of the network) that compose the sensors network to communicate the start of activities. When a specific event is detected, the involved smart Web sensor sends an asynchronous packet to the supervisor node in which is reported the time stamp and the typology of the event. The supervisor node stores the information and begins to download the data form every sensor present in its sensors network. At the end, all the saved data are accessible via the Web service interface.

3. THE DEVELOPED DISTRIBUTED MEASUREMENT SYSTEM

3.1 The developed Web Service UDDI Access

A personal UDDI registry has been implemented to develop the distributed measurement system. It has been used jUDDI project ver.2.0 because it is an open source Java implementation of the UDDI specification for Web services.

To implement the UDDI registry by jUDDI, on a local personal computer with Fedora Core 4 has been installed: i) Java 2 SDK - used Sun's Java 2 SDK SE, version 1.4.2_04; ii) J2EE build and runtime environment - used Sun Java 2 Enterprise Edition (J2EE) 1.3.1; iii) Web server and/or servlet container - used Apache Tomcat, version 5.0.24; iv) SOAP processing framework - used the version of Apache Axis that ships with jUDDI; v) Data storage mechanism - used the MySQL relational database, version 4.0.19; vi) UDDI registry framework - the registry framework jUDDI

3.2 The developed smart UDDI sensor

The smart UDDI sensor is an embedded system able to perform measurement tasks and to communicate over the Internet network. It has been adopted only open source software for the developing of the distributed measurement system. As a multipurpose operating system, Linux is used for a wide variety of purposes including networking, software development, as well as an end-user platform. Apache is an implementation of an HTTP server and it is the most popular Web server in use today. MySQL is an implementation of a database server that is known for its speed and reliability. PHP is general-purpose scripting language that is particularly suited to Internet-based system development and is the most widely used Apache module. The hardware architecture is based on a motherboard that is made operative via a light Linux distribution (Damn Small Linux). The distribution has been installed on a USB pen driver, while the database is stored in another USB mass storage drive to separate the kernel with the data. A diagram block of the whole system is reported in the Figure 3, while in the Figure 4 is reported a photo of the realized prototype.

The proposed smart Web sensor has been adopted for a power quality monitoring application. The Linux embedded board performs some tasks: i) it receives information from the Fluke 45 via the RS-232 port; ii) the C program saves the received data in a database mounted on a USB mass storage device; iii) the Web server manages a PHP Web service for the publishing on demand of the present services through the network interfaces connected to the system; iv) it communicates with the UDDI registry for the discovering of the services; v) it communicates with the client part of the WSUA.

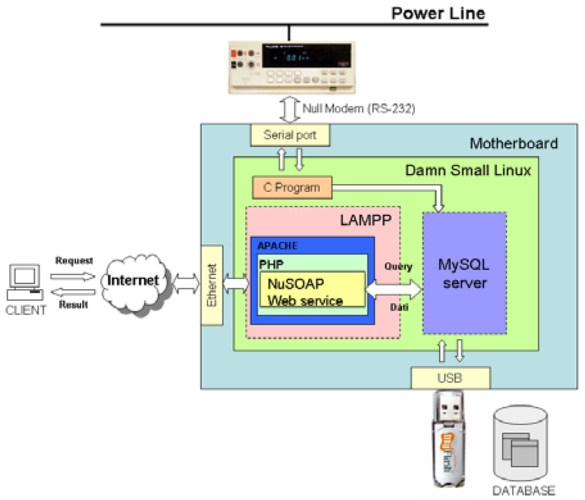

Fig. 3. Hardware and software architecture of the smart Web sensor

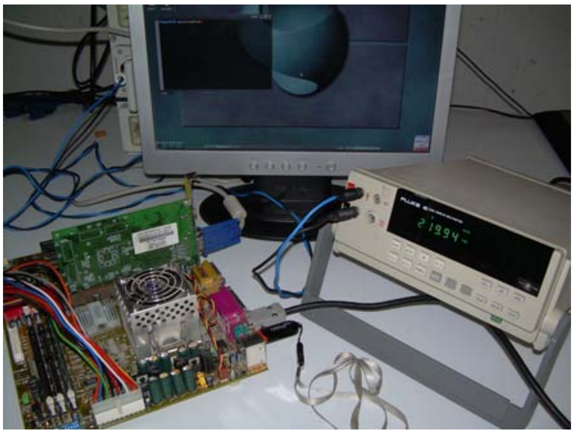

Fig. 4. The developed smart UDDI sensor

The power line is measured by a multimeter Fluke 45 which performs True-RMS voltage measurement with an accuracy of 0.05% and a bandwidth of 1 MHz. The Fluke 45 is connected to the motherboard via a RS-232 interface. A low level program, written in C, performs same tasks: i) managing the acquisition timing, ii) quering the Fluke 45 to perform a measure; iii) parsing the string which contains the measure in a floating number; iv) saving the new data with the time stamp in the daily table; v) performing the evaluation of maximum, medium and minimum at every new value so, when the pointer of current value has performed a complete cycle in the daily table, the data can be saved in the historic table.

3.3 The proposed UDDI interface

For the proposed smart Web sensor it has been developed a UDDI interface to standardize the input format, the output format and the methods [16]. Every format is stored in a UDDI tModel. In particular, two Web methods have been developed:

1. GetDailyRMS(start_time,stop_time): with this method a client can perform a request to download the daily data from the start time to the stop time from the smart Web sensor . The output is an array of items that presents the stored value and the time when the event is occurred.
2. GetHistoricRMS(start_date,stop_date): with this method a client can perform a request to download the historic data from the start date to the stop date from the smart Web sensor . The smart Web sensor provides some processing before to store the data into the local database. In particular, it has been performed the Maximum, Medium and Minimum processing. The output is an array of items in which every item is a collection of three sub-items. Every sub-item presents the Maximum, Medium or Minimum value and the date and time when the event is occurred.

3.4 The Supervisor node

As reported in Figure 2, the supervisor node covers an important rule in the sensors network. It manages and controls the smart Web sensors (SWS) and it interfaces the functionalities exported by the sensors (WS) to the clients connected to the network. So, every sensors network has refereed to a particular supervisor to create a scalability of the network management and to identify the sensors network with its master (the supervisor). To give much more dynamicity and flexibility to the sensors network the supervisor node use two ways to communicate with the clients and with the smart sensors: i) during a normal access, it routes the client request directly to the sensor involved in the communication; ii) if a sensor notices an abnormal behaviour of system under test, it uses a symmetric asynchronous technique to communicate the event to the supervisor node [17]. In Figure 5 is reported the technique adopted that is similar to the AJAX technology: the supervisor node sends a not-blocking request to every smart sensor connected to it and continues its activities until a response is received. If an abnormal event has noticed, the supervisor node requests the measured data before and after the event from all the sensors connected to it for having a global view of the phenomenon. The data are stored locally in a database present on the supervisor node for an historical archive. The clients connected to the supervisor node can access to the archive using the Web service technology or, if they are connected during an abnormal event recorded by the supervisor node, the data are immediately sent.

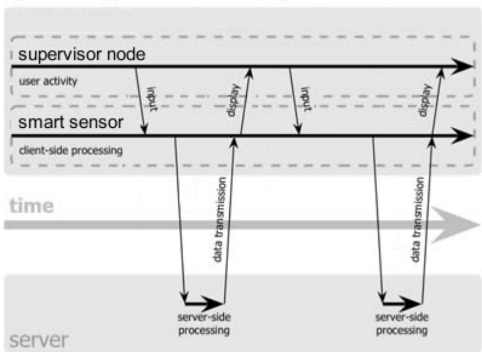

Fig. 5 The symmetric asynchronous approach used by the supervisor node

Fig. 6. A screenshot of Client GUI during a manual access

Fig. 7. A screenshot of Client GUI during an asynchronous event

4. THE EXPERIMENTAL RESULTS

The whole system has been tested to verify the proposed architecture. Using the Web service UDDI access, a client has been developed with Visual Studio 2005 .NET to consume the proposed smart sensor. The client has been written in VB .NET using the Framework .NET 2.0. The Windows form allows the user to access to the smart Web sensor directly or via the supervisor node to download the data from the daily table and the historic table. Selecting the period of time from the ActiveX control and push on the Download Data, the Windows form access to the supervisor node which download the data from the smart sensor. On the right side of time-depend graphs, a static histogram of all downloaded data has been developed to give much more information to the user. The histogram divides the data in eleven classes for the daily RMS values and in seven classes for the historic RMS values. The frequencies of every class are evaluated and then plotted on the user interface.

In the Figure 6, a typical RMS behaviour has been presented during a manual download. The time to download 50 floating point data is 0.546598 s. The time measurement test has been provided using the software Ethereal (network analyzer software). In the Figure 7, a RMS step variation has been reported during an asynchronous event. The connection time is 0.230876 s while the download time is 0.426716 s. In this test only the RMS behaviour has been downloaded, while the Historical data are unchanged.

5. CONCLUSIONS

In this paper a new architecture of an event-based asynchronous sensors network has been presented. The architecture adopts a hierarchical approach to control and to manage the remote smart sensors by means of a supervisor node. Every supervisor node identifies a particular sensors network and uses a symmetric asynchronous communication technique to consume the smart sensors services and to exchange data with the client connected to the network. The sensors network uses the tModel description to create a common interface to publish the services and to create taxonomies of accessible services. To test the proposed network, a smart sensor using only open source and freeware software for a PQ monitoring application has been developed. The smart sensor is based on embedded Linux distribution mounted on a USB pen drive while the acquisition part uses a Fluke 45 multimeter connected with the system via the RS-232 interface. To give much more freedom to the whole system, a new Web service for UDDI access has been developed to create an interface between the client and the UDDI registry and between the client and the smart Web sensor. In this way, the client can see a Web service of Web services and can found the specific Web service only with standard Web method without knowing UDDI APIs. A user interface based on Framework .NET 2.0 has been developed to consume the network. The results show the client can access to the remote DB present on the smart Web sensor to download the measurement data. Moreover, the supervisor node can communicate in asynchronous way to the remote client and to the smart sensor to reduce the waiting time and to avoid the polling request/response. At the end, the measured connection and download times show a good response of the proposed approach.

REFERENCES

[1] C. Muscas, L. Peretto, S. Susil, R. Tinarelli “Implementation of multi-point measurement techniques for PQ monitoring”, in Proc of IEEE Instrumentation and Measurement Technology Conference, Como, Italy, 18-20 May 2004.

[2] C.Muscas, L. Peretto, S. Sulis, R. Tinarelli “Effects of load unbalance on multi-point measurement techniques for assessing the responsibility for PQ degradation”, in Proc. of IEEE 11th International Conference on Harmonics and Quality of Power, 12-15 Sept. 2004.

[3] F. Ciancetta, B. D’Apice, D. Gallo, C. Landi, “Architecture for Distributed Monitoring based on Smart Sensor and Web Service”, in Proc. of IEEE Instrumentation and Measurement Technology Conference, Sorrento, Italy, 24-27 April 2006.

[4] F. Ciancetta, E. Fiorucci, B. D'Apice, C. Landi, “A Peer-to-Peer Distributed System for Multipoint Measurement Techniques”, in Proc. of IEEE Instrumentation and Measurement Technology Conference, Warsaw, Poland, May 1-3, 2007.

[5] Kang Lee and Eugene Song, “Smart Transducer Web Services Based on the Proposed IEEE 1451.0 Standard”, in

Proc. of IEEE Instrumentation and Measurement Technology Conference, Warsaw, Poland, May 1-3, 2007.

[6] F. Ciancetta, B. D'Apice, C. Landi, A. Pelvio "Mobile Sensor Network Architecture for Environmental Monitoring", in Proc. of IEEE IMEKO TC-19, Iasi, Romania, September 19-21, 2007.

[7] Tomasz Mielcarz and Wieslaw Winiecki, "The Use of Web-services for Development of Distributed Measurement Systems", in Proc. of IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 5-7 September 2005 ,Sofia, Bulgaria, pp 320–324.

[8] F. Ciancetta, B. D'Apice, D. Gallo, C. Landi, "Plug-n-Play Smart Sensor Based on Web Service", IEEE Sensors Journal, 7, issue 5, May 2007, pp 882–889.

[9] Jiamao Liu, Ning Gu, Yuwei Zong, Zhigang Ding, Shaohua Zhang and Quan Zhang, "Service Registration and Discovery in a Domain-Oriented UDDI Registry", in Proc. of the Fifth International Conference on Computer and Information Technology (CIT'05), Shanghai, China, 21-23 September, 2005, pp 276–283.

[10] OASIS, UDDI Registry tModels, Version 2.04, Available at http://uddi.org/taxonomies/ UDDI_Registry_tModels.htm

[11] S.M. Pahlevi, H. Kitagawa, "Taxonomy-based adaptive Web search method", in Proc. of International Conference on Information Technology: Coding and Computing, Las Vegas, NV, USA, 8-10 April, 2002, pp 320–325.

[12] OASIS, Providing a Taxonomy for Use in UDDI Version 2, Available at http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-tn-taxonomy-provider-v100-20010717.htm

[13] OASIS, UDDI Spec Technical Committee Draft, Dated 20041019, Available at http://www.oasis-open.org/committees/uddi-spec/doc/spec /v3/uddiv3.0.2-20041019.htm.

[14] OASIS, UDDI Version 2.04 API, Published Specification, Dated 19 July 2002 Available at http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.pdf.

[15] Jian Jun Yu, Gang Zhou, "Dynamic Web Service Invocation Based on UDDI", in Proc. of IEEE International Conference on E-Commerce Technology for Dynamic E-Business (CEC-East'04), Beijing, China,13-15 September, 2004, pp 154–157.

[16] Guan Chao, Li Xiujun, Meijer G.C.M., "A system-level approach for the design of smart sensor interfaces", Proceedings of IEEE Sensors, 2004, Vienna, Austria, 24-27 October, 2004, pp 210–214.

[17] R. van Glabbeek, U. Goltz, J.-W. Schicke, " Symmetric and Asymmetric Asynchronous Interaction". Proc. of 1st Interaction and Concurrency Experience (ICE'08) Synchronous and Asynchronous Interactions in Concurrent Distributed Systems, Reykjavik, Iceland, 6th of July 2008.