Journal of Machine Engineering, Vol. 14, No. 2, 2014

communication,
manufacturing system,
process monitoring, decentralisation

Jeff MORGAN[1]
Garret E. O'DONNELL[1]

A SERVICE-ORIENTED RECONFIGURABLE PROCESS MONITORING SYSTEM – ENABLING CYBER PHYSICAL SYSTEMS

Communication is an essential requirement for collaborative manufacturing systems. However the diversity of communication media and protocols in machine tools, automation equipment, and associated proprietary software, presents a challenge for enabling capable, extensible and re-configurable process monitoring systems. Additionally, as process control systems evolve from centralised hierarchical structures to decentralised heterarchical communities, enabling media and tools are required to provide interoperability between systems and subsystems. The focus of this research is to introduce a manufacturing decentralised process monitoring architecture that utilises a service-oriented architecture framework for network-wide dynamic data acquisition and distribution. The system design is created using a combination of service-oriented architecture topology and technical modelling. Service-oriented communication structure and capability is given particular focus, resulting in a comparative study of message structures and communication speeds. The resultant system is modular in structure, reconfigurable, network-distributable, interoperable, efficient, and meets real-time requirements.

1. INTRODUCTION

The performance of complex manufacturing structures ultimately hinges on their ability to rapidly adapt their production to current internal and external circumstances [1]. Today's turbulent market place has a movement towards high product mixture and low product volume production [2]. Manufacturers need to have the ability to be cooperative and have a quick response to market changes and disturbances in order to stay competitive [1]. Similar to the production technology, production control and monitoring systems have moved away from central operational structures and towards Decentralised Control Systems (DCS) [3]. The introduction of intelligent and reconfigurable, or adaptable manufacturing systems, with a modular architecture which can be restructured without a loss in efficiency, has defined a shift in the manufacturing technology paradigm, that is aimed at enabling the manufacturing plant of the future [4].

Service Oriented Architecture (SOA) has been at the forefront of industrial research and development into DCS, from the interconnected EU funded projects SIRENA, SODA,

[1] Mechanical and Manufacturing Engineering Department, Trinity College Dublin, Dublin 2, Ireland
E-mail: morganje@tcd.ie

SOCRADES, and AESOP [5]. A SOA is a set of architectural tenants for building autonomous yet interoperable systems [6]. SOA specifies that distributed resources and organisations should provide their functionalities in the form of services, which requesters can have access to [7]. An entity or 'Service' only exposes its interface which can be discovered dynamically and allow for asynchronous messaging [8]. In doing so SOA systems enable multiple client-oriented entities to utilise the resources embedded within the service, making the way for more reconfigurable and flexible decentralised systems. The culmination of SOA research resulted in a comprehensive DCS architecture defined within the AESOP architecture. This provided a cloud of manufacturing services with the potential of facilitating the requirements of an entire manufacturing enterprise (Fig. 1). AESOP utilises Device Protocol for Web Services (DPWS) as a central architecture for device networking [9]. However this data interoperability standard is not domain-specific and utilises an openly interpreted XML meta-model representation of resources. This means that the protocol represents a comprehensive medium of data access and transfer from device to device, while the application specific data and services being provided are within a user defined custom model structured with XML. The DPWS protocol has been experimented with for industrial use within DCS in AESOP [10].

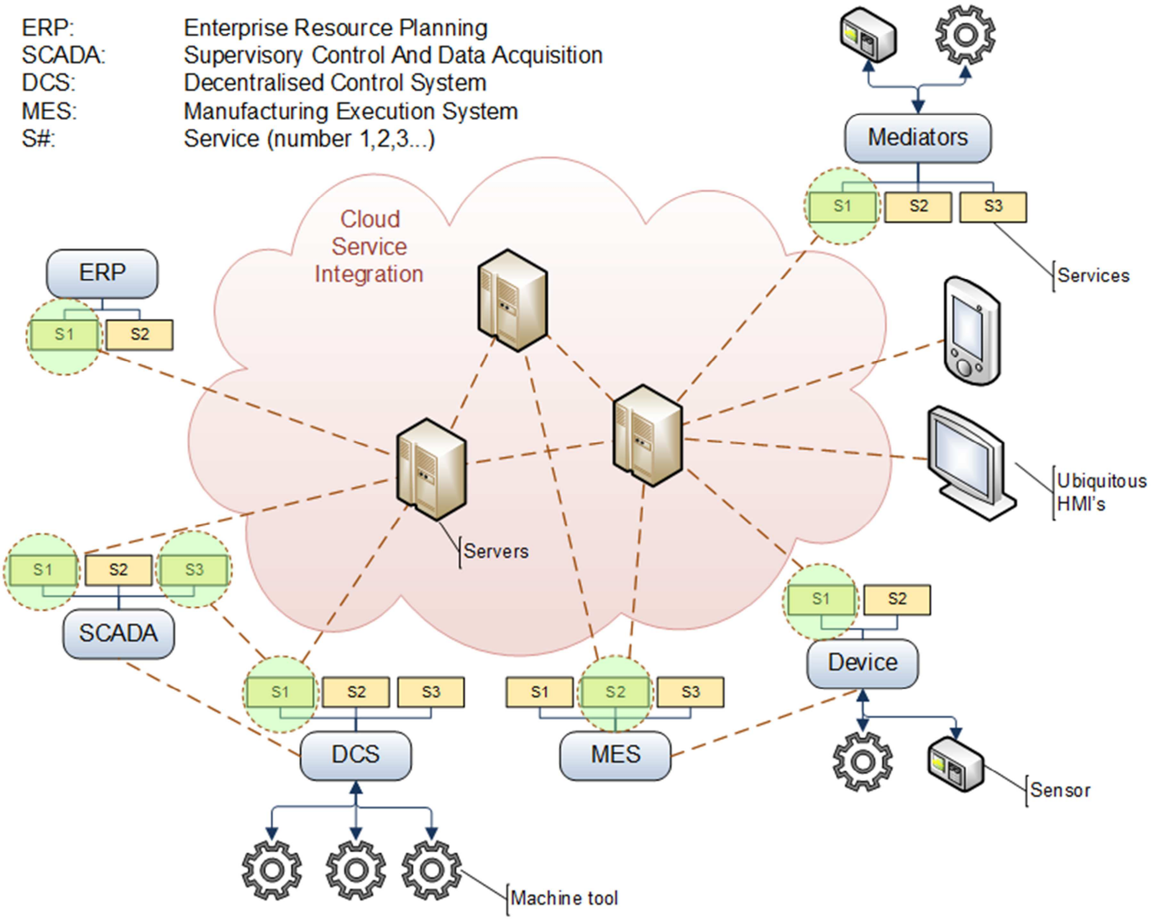

ERP: Enterprise Resource Planning
SCADA: Supervisory Control And Data Acquisition
DCS: Decentralised Control System
MES: Manufacturing Execution System
S#: Service (number 1,2,3...)
Cloud Service Integration
Mediators
S1
S2
S3
Services
ERP
S1
S2
Servers
Ubiquitous HMI's
S1
S2
S3
SCADA
S1
S2
Device
S1
S2
S3
DCS
S1
S2
S3
MES
Sensor
Machine tool

Fig. 1. AESOP architecture, adapted from [11]

However the meta-models are not reviewed, nor are any examples provided outside the scope of traditional Supervisory Control And Data Acquisition (SCADA) system implementation, e.g. use within high speed/sample rate Process Monitoring Systems (PMS).

The focus of this work is to define a topology for Reconfigurable Process Monitoring Systems (RPMS), through exploring how SOA can be utilised within a manufacturing PMS. Key process monitoring requirements are explored; data transmission speed, data structuring, data compression, architecture characterisation and configuration. A novel SOA RPMS is defined that utilises a unique data interoperability technology and compression message structure, to enable efficient dynamic data acquisition and distribution across a network.

2. SERVICE ORIENTED RECONFIGURABLE PROCESS MONITORING

A reconfigurable manufacturing system is designed at the outset for rapid changes in structure, as well as in hardware and software components, in order to rapidly adjust production capacity and functionality within a part family in response to sudden changes in market or in regulatory requirements [12]. Within the field of manufacturing, SOA offers the potential to provide the necessary system-wide visibility and device interoperability for complex collaborative automation systems [13]. The incorporation of SOA within a PMS will enable a reconfigurable system of interactive components. These components can then be utilised in a multitude of manufacturing systems, as their flexible nature will enable adaption to new processes. Additionally, as manufacturing systems can be reconfigurable in nature a RPMS can allow for this variation change with the process.

2. 1. RECONFIGURABLE PROCESS MONITORING SERVICE ORIENTED ARCHITECTURE MODELLING

In 2012 Mora et al. [5] reviewed the research and development of the interconnecting EU funded projects SIRENA, SODA, SOCRADES, and AESOP. This work shows the evolution of SOA within the field of manufacturing DCS, from interoperability between embedded devices within the SIRENA project, to cross layer service collaboration to meet the needs of a manufacturing enterprise within the AESOP project. This research defined two fundamental models for creating a SOA; an engineering topology model and technical data model. An engineering approach to SOA incorporates a five method model aimed at creating a SOA topology of a desired system [14]. The five method steps are; legend, domain and system categorisation, interface definition, service and orchestrator integration, and topology generation. An SOA data model abstracts a SOA into a common structure that can be described by four technical layers; Meta Model, Data Model, Generic Services, and Mapping on Protocols [15]. These two models have similar aspects, yet they have a unique perspective for defining an SOA. In this article a cross-over modelling approach is applied consisting of the following steps; Legend Definition, Domain Specification, Meta and Data Modelling, Service and Mapping, and Topology Generation.

2. 2. LEGEND DEFINITION

A legend identifies the symbols for different SOA elements across multiple domains. These symbols include inputs, outputs, boundary layers, and events [14]. These components define the different objects present within the SOA. A recent CIRP keynote paper Teti et al. [16] characterised process monitoring into the following steps; (a) Measurement: physical hardware, e.g. sensors, for measuring the physical process parameter; (b) Acquisition: interconnecting hardware and software elements for providing high speed data acquisition from the sensor to a computational device; (c) Filtering: mathematical manipulation of data for specific process feature extraction; (d) Analysis: methods, techniques and algorithms for variable correlation of required process attributes; (e) Decision Support: subsequent methods, techniques and algorithms appertaining to identifying the required corresponding process action from analysed results; (f) Closed-loop control: hardware and software elements associated with facilitating corrective action from decision support functions.

An RPMS's elements can be categorised utilising these steps, Fig. 2.

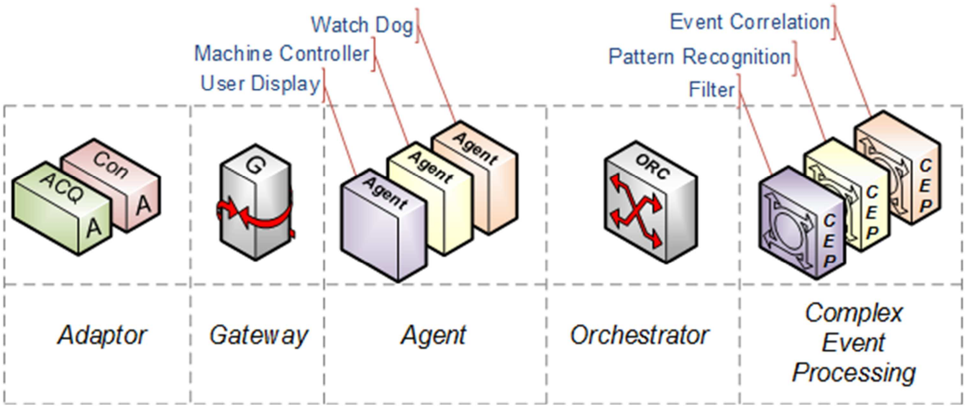


Fig. 2. Reconfigurable process monitoring system legend

- Adaptor: the physical connection points of cyber-physical-systems [17], which can acquire data from a physical or virtual source and release the data within the SOA as a service, or oppositely transfer system commands or data to a physical controlled body. Adaptors would represent data acquisition and closed loop control steps within a PMS. Adaptors are either Control Adaptors, or Acquisition Adaptors.
- Complex Event Processing (CEP); entities that derive and analyse higher level information out of low-level or atomic events [18]. The scope of CEP is limitless due to its open definition as any data manipulation services can be represented as a CEP engine. A defining characteristic of a CEP element is its ability to acquire data from a source, process it, and then re-release it as a new service to the network. CEP can represent the filtering, analysis, and decision support steps within a PMS. CEP is an open representation of a computational service, and can be distinguished by their primary functions, e.g. CEP-Filter, CEP-Analysis, etc.

- Agents; active consumers of service data for individual utilisation. The scope of Agent entities is limitless due to its open definition. This is because any data consumer can be represented as an Agent. A defining characteristic of an Agent is its localised individual use of data, as it does not release data back into the system as a service. Agents if required can utilise multiple services that are available within the network, communicate with entities to achieve goals, and perform system control actions through the available services. Agents can represent the filtering, analysis, decision support, and closed loop steps within a PMS. Agent is an open representation of a system component, and can be distinguished by their primary functions, e.g. Data Display-Agent, Filter Agent, Management-Agent, etc.

System management and control functional elements explored within the DCS domain include;

- Gateway/Mediator elements: enable the connection of different network types within the architecture, or provide a means of transportation of data to different network areas for distribution [19].
- Orchestration elements: central control applications which can dictate operation to organise decentralised entities, or enable interoperability between two or more entities [20].

2. 3. DOMAIN SPECIFICATION

Domain and system categorisation incorporates the assembly of domain-specific components and categorisation within layers defined in the ISA 95 standard [14]. ISA 95 is an international standard that was created to define models and terminology to determine which information has to be exchanged between systems for sales, finance and logistics and systems for production, maintenance and quality [19]. Level 0 is the production process itself; Level 1 is associated with all sensing and manipulating elements within the production process; Level 2 addresses monitoring, supervisory control, and automatic control of the production process; Level 3 incorporates the management of the workflow to produce the desired end-products, maintaining records and optimising the production process; finally Level 4 aims at establishing the basic plant production schedule, material use, delivery and shipping, and inventory.

The focus of this work is within level 1 and level 2, demonstrating the value of enabling SOA in sensing and monitoring within a PMS. The capability of SOA DCS to achieve enterprise wide integration throughout higher levels is self-evident in the AESOP project. However, this work aims at providing a specific model starting with a bottom-up approach to RPMS, consisting of the Measurement and Acquisition steps within a PMS. An example of which can be seen in the merging of infrastructure layers across domains, Fig. 3. Each domain requires monitoring of the manufacturing process, this data can be subsequently shared by providing its action as a service to applications inside and outside the specific domain of initial implementation. The SOA Adaptor element can enable this functionality by incorporating a unique data acquisition functionality that takes inputs from a sensor to producing an output that is hosted as a service and transmitted to multiple sources.

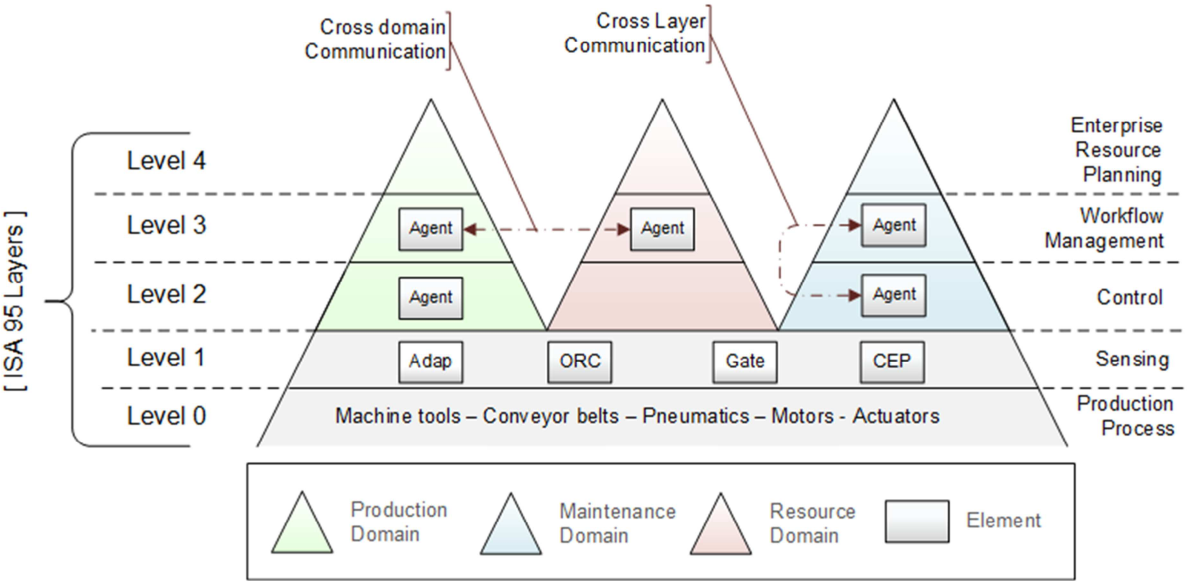


Fig. 3. ISA-95 layers [14]

2. 4. META AND DATA MODELLING

A Meta Model defines the basic components that the data model can be built from which includes concepts and rules. A Data Model is a semantic or abstract description of the data owned by a subsystem which can be accessed by other systems within a specific domain [15]. Within the Adaptor element, the data model can be seen to represent different types of data that is able to be acquired by data acquisition functions present within the application, Fig. 4. This data has sub-type-data corresponding to different variable parameters, i.e. its data type, sample rate, unit representation, and time of occurrence. Timing within a process monitoring system is crucial as it enables the correlation of different data streams through an instance of occurrence reference. This requires data that is being acquired to share a timing element, e.g. a clock. Traditional SCADA systems are less concerned with correlation, given that their requirements are specific to the most recent data reading. However, sampling rates are dependent on the resolution of measurements required for signal analysis. High sampling rates provide a better representation of the signals behaviour. Low sampling rates can miss important behavioural occurrences and potentially cause signal aliasing. Signal aliasing produces an incorrect representation of signals state, as high frequency signals can be aliasing as low frequency signals due to a low sample rate. Subsequently sample rate requirements within PMS can range between 1Hz to 1MHz and beyond [16]. Computational-units or networks have an incapability to sustain such a large amount of traffic, especially for single value references at high frequency. To overcome this challenge data is grouped into packets, with meta-data specifying time of occurrence within the reference clock. Within the data model, specified in Fig. 4, a data packet is a collection of raw data points with timing-meta-data, namely; Clock: specifies a reference point from the acquisition clock appertaining to when data was first acquired; Time-Line: the total time that has passed since the initial clock reading was taken; T-Delta: the common time increment between samples which is dictated by the sample rate. The combination of raw-data and timing-meta-data enables a data stream to be packed by a sourcing application and then subsequently assembled and consumed by a seeking application.

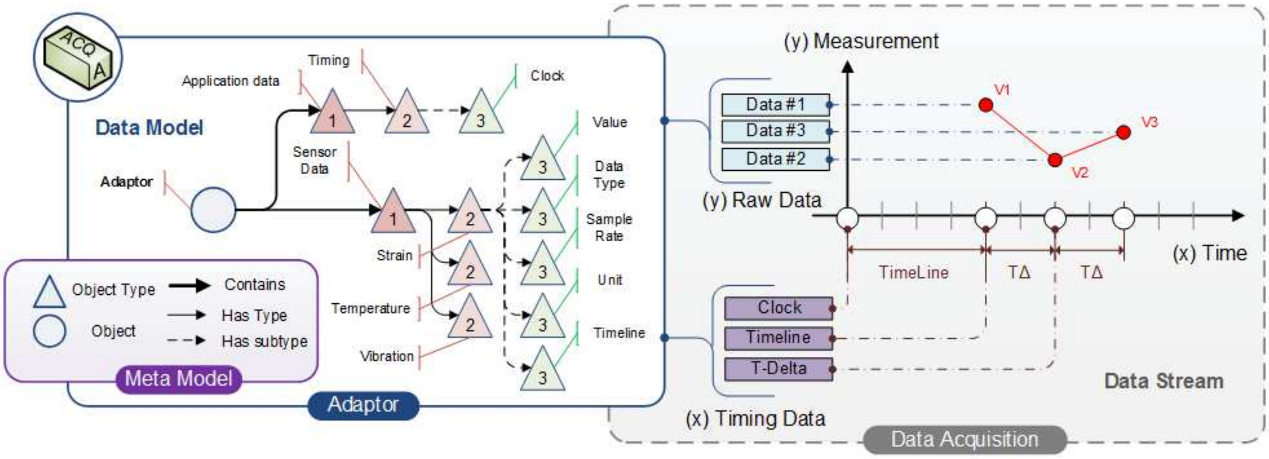


Fig. 4. Adaptor modelling

2. 5. SERVICES AND MAPPING

Generic Services are an abstract common way for exchanging data between subsystems that are technologically independent; and Mapping defines how abstract services are mapped for physical implementation [15]. Previous work [21] [22] has shown how data DPWS provides services and physical mapping to enable data interoperability within a network Hosting services provided by DPWS include messaging, discovery, description, and eventing [23]; Messaging services provide the transmission of messages between systems; Discovery services are used by a device connected to a network to advertise itself and to discover other devices; Description services incorporates metadata exchange to provide information about a device and the hosted services on it; Asynchronous publish and subscribe eventing allows multiple systems to subscribe to asynchronous event messages produced by a given hosted service. The defined DPWS utilises a XML data structured service model that has the benefit of providing great flexibility to application developers and enhance interoperability. However this brings high overhead in terms of memory, CPU, latency and power [24]. The DPWS performance was characterised in the

SOCRADES project, as the total time required to send a message/event from one device to another over a Local Area Network (LAN) [25]. Results show a communication speed of 10ms, with overall mechanical system reaction time in the range of 100ms to 1s, which is considered by end-users as insufficient for some solutions. Subsequently text-based XML WS were identified to not meet real-time requirements and resource constraints for industrial machinery applications. In order to meet industrial requirements, the research performed within the scope of the AESOP project has identified the potential present within the Efficient XML Interchange (EXI) standard to overtake traditional XML structuring within DPWS [26]. EXI is a binary representation of the XML information set that is designed for compactness and high performance parsing and serialisation. Binary representation, within interoperability systems pose significant improvements in resource utilisation, e.g. network throughput, computational power and memory.

2. 6. SHARED VARIABLE ENGINE

The authors wanted to provide the common services outlined within the DPWS for dynamic data acquisition and distribution, within the development of the RPMS. However, the authors also wanted to incorporate the use of binary conversion methods to structure and compress data for efficient data transfer. In order to meet these requirements a SOA technology was identified that incorporates data interoperability, discovery, and eventing services; namely the National Instruments Shared Variable Engine (SVE). The SVE is a software framework that enables variables to exist on a network and be communicated between applications, remote computers, and hardware [27]. The SVE utilises the NI-Publish Subscribe Protocol (NI-PSP), which utilises Ethernet TCP/IP and a LogosXT transmission algorithm [28]. The SVE enables applications to expose their data as services, by ‘publishing’ the data to a SVE. The SVE hosts the data, buffers the data, and distributes the data to multiple applications which ‘subscribe’ to it.

2. 7. BINARY MESSAGE MODEL

To distribute data within a network via the SVE a message structure needs to be defined. The requirements for the message structure includes; multi-sample, the message must be capable of containing a single variable value as well as multiple values; meta-data, the message requires the incorporation of timing-meta-data and variable characterisation data; data-types, the message must be able to incorporate multiple data types, e.g. Boolean, Double, Integer, etc; binary-compression, the raw data and message must be compressed into a binary representation. A three step Binary Message Model (BMM) was utilised to meet the previous stated requirements, Figure 5. Binary conversion enables the conversion of any data type or cluster to a binary string format which can be represented as byte string. Both the process-data and metadata could be converted into byte strings and structured in a generic message model, which identifies what type of message it is, the owner of the message, and the time at which the message was created.

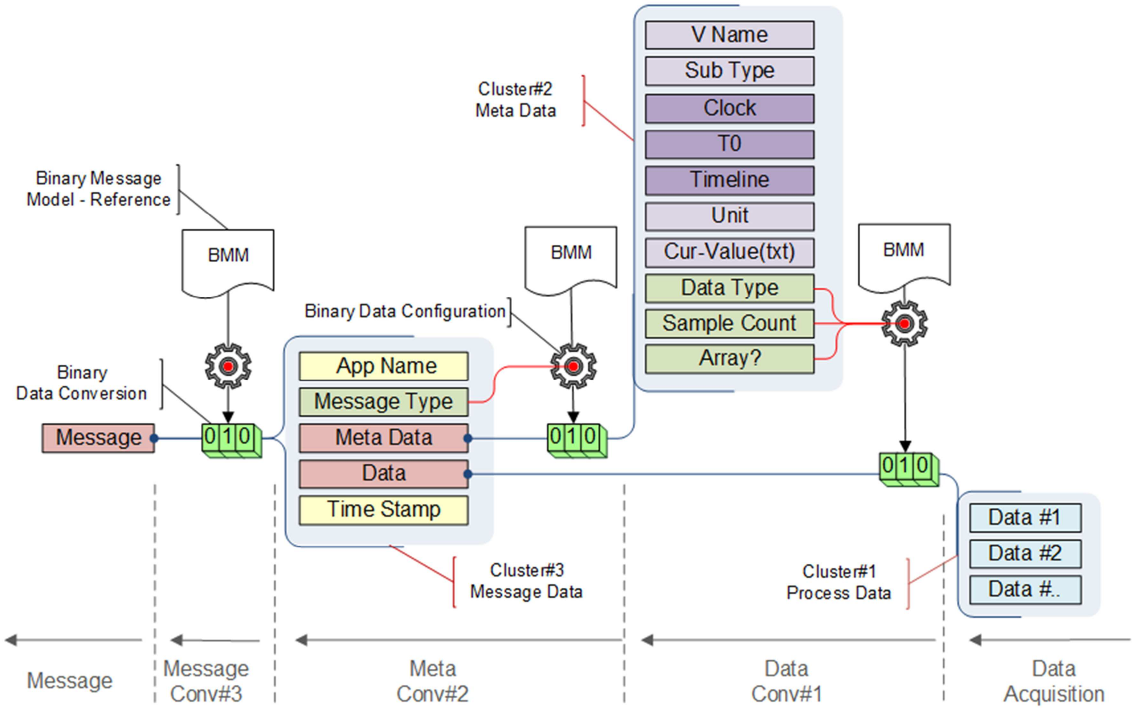


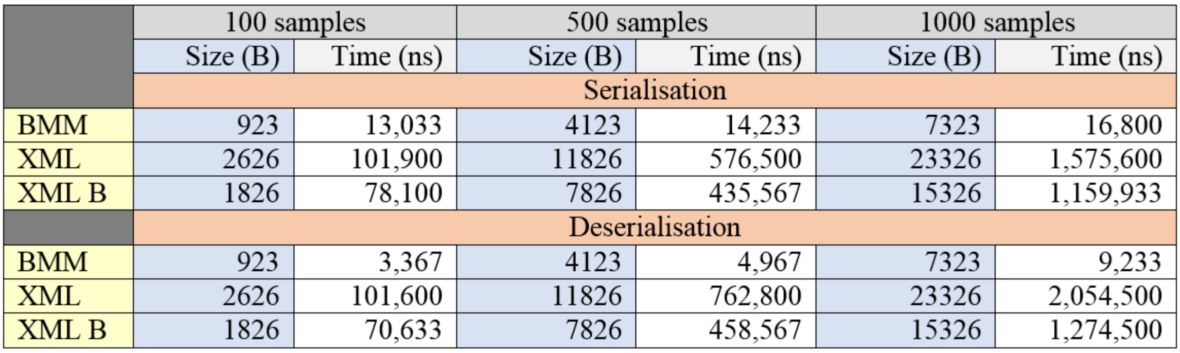
Fig. 5. Binary message model

Binary conversion enables flexibility within the message as different data types and quantities can be assembled without the need for changing the message structure. In order to decode the message the receiving application needs a data structure reference in which to transpose the binary data into. The initial message data model acts as a base reference model to achieve this, enabling the message type to be exposed and enabling the application to determine what structure the metadata is in. Subsequently, the metadata will provide details in how to transpose the process data binary string, e.g. data-type, sample count, and whether or not it is an array of data. The BMM can be expanded to incorporate multiple types of messages, including requests and response messages between applications. The only change will be present within the message type specified and the corresponding data structure of the metadata and data binary strings.

2. 8. PERFORMANCE MEASUREMENT

To test the performance of the SVE and BMM, the message structure serialisation and deserialisation time frame was quantified in addition to Round Trip Time (RTT). RTT measures the time taken to send a message to a networked application and receive the same message. Three types of message structure were tested to enable a performance comparison; the BMM, normal XML coding, and a hybrid structure which utilised binary conversion of sample data values with a XML structure. The data contained within the messages was in accordance with the BMM, consisting of metadata and raw data, which had a varying data quantity of 100, 500, and 1000 samples per message. All three methods utilised the SVE as an interoperability medium. The SVE enables data interoperability across a network and within a central computer. Both scenarios were tested for RTT capabilities. Experimental results are given in Table 1 and Table 2. Karnouskos and Somlev [29] (Table 3) performed similar experiments in order to assess the performance of WS, namely; traditional web services with Axis2, DPWS, REpresentational State Transfer (REST), and Constrained Application Protocol (CoAP). A comparative analysis of results from both studies is given in Fig. 6.

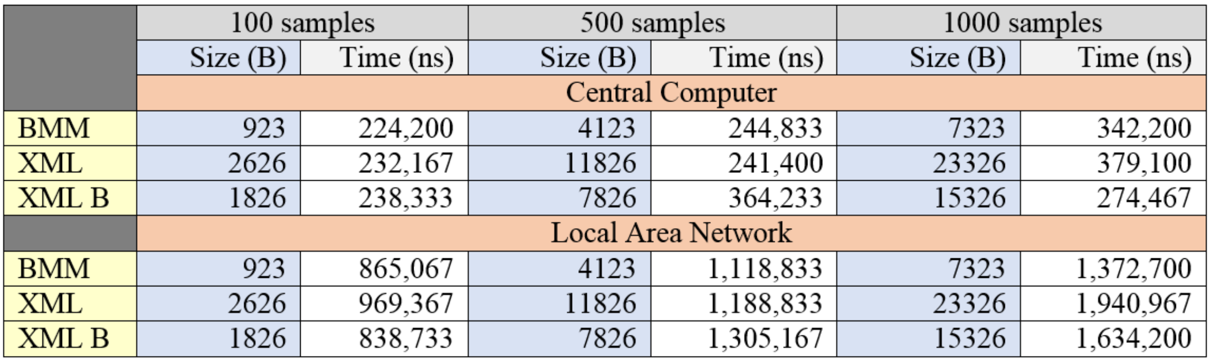
Table 1. SVE Experimental results: serialisation and deserialisation



	100 samples		500 samples		1000 samples	
	Size (B)	Time (ns)	Size (B)	Time (ns)	Size (B)	Time (ns)
	Serialisation					
BMM	923	13,033	4123	14,233	7323	16,800
XML	2626	101,900	11826	576,500	23326	1,575,600
XML B	1826	78,100	7826	435,567	15326	1,159,933
	Deserialisation					
BMM	923	3,367	4123	4,967	7323	9,233
XML	2626	101,600	11826	762,800	23326	2,054,500
XML B	1826	70,633	7826	458,567	15326	1,274,500

*All data points consisted of double precision floating point @ 8 Bytes; values = 123456789.123456
* Time values represent the mean 3 test consisting of 1000 measurements each

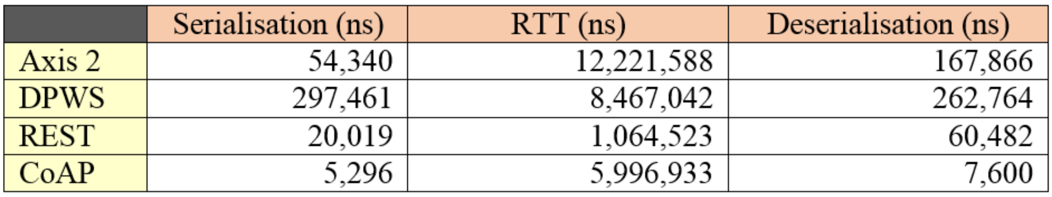
Table 2. SVE Experimental Results: Round Trip Time (RTT)



	100 samples		500 samples		1000 samples	
	Size (B)	Time (ns)	Size (B)	Time (ns)	Size (B)	Time (ns)
	Central Computer					
BMM	923	224,200	4123	244,833	7323	342,200
XML	2626	232,167	11826	241,400	23326	379,100
XML B	1826	238,333	7826	364,233	15326	274,467
	Local Area Network					
BMM	923	865,067	4123	1,118,833	7323	1,372,700
XML	2626	969,367	11826	1,188,833	23326	1,940,967
XML B	1826	838,733	7826	1,305,167	15326	1,634,200

*All data points consisted of double precision floating point @ 8 Bytes; values = 123456789.123456
*Time values represent the mean 3 test consisting of 1000 measurements each

Table 3. Karnouskos and Somlev experimental results [29]



	Serialisation (ns)	RTT (ns)	Deserialisation (ns)
Axis 2	54,340	12,221,588	167,866
DPWS	297,461	8,467,042	262,764
REST	20,019	1,064,523	60,482
CoAP	5,296	5,996,933	7,600

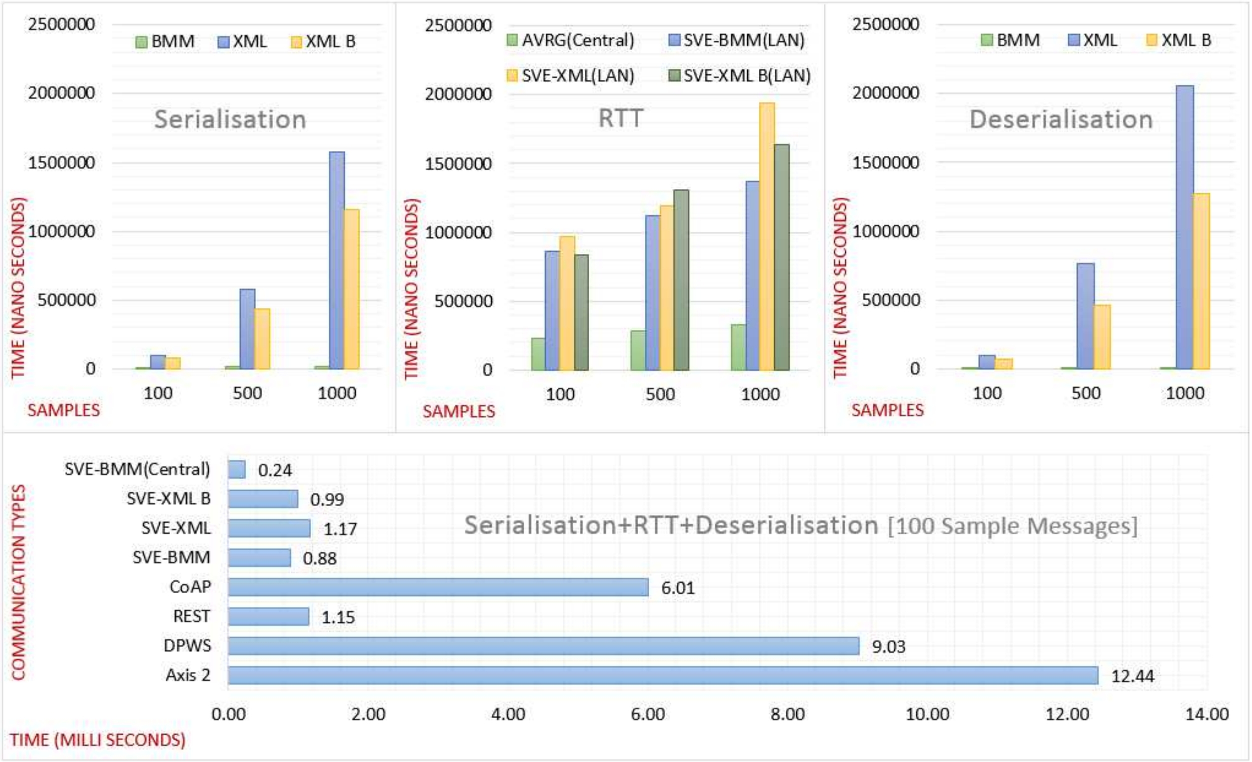


Fig. 6. Shared variable engine experimental results

The results identify an 87.2% time reduction when utilising the BMM serialisation at 100 samples compared to SVE-XML and a 95.6% reduction compared to DPWS. Data compression within the BMM produced on average a 66.2% reduction in message size compared to XML. BMM deserialisation provided a 96.7% time reduction at 100 samples compared to SVE-XML and a 98.7% reduction compared to DPWS. The SVE RTT has a linear response to data transmission size. This provided a reduced RTT of 89.1% on average at 100 samples per messages across SVE BMM, XML, and XML B compared to DPWS. Comparatively, the SVE in combination with the BMM provides the shortest serialisation, RTT, and deserialisation time of all measured services at 0.88ms, with a sample rate of 100K Hz at 100 samples per message, within a LAN, which is within the 1ms requirement of real-time systems. These results also indicate that local interoperability within a central computer can yield greater time reductions as the SVE provided a 0.23ms RTT. The combination of SVE and BMM within a central computer can provide interoperability within 0.24ms with a sample rate of 100K Hz at 100 samples per message, or 0.37ms with a sample rate of 1 M Hz at 1000 samples per message.

2. 9. TOPOLOGY GENERATION

Topology generation connects all components within domains, and between domains, using the previously defined interfaces creating a topology of the SOA system. A simple

RPMS topology for data acquisition, filtering, and analysis can be seen in Fig. 7. Adaptors acquire data and publish it to the SVE. Connected computers can gain access to the data via local SVE's that provide for network interoperability. CEP elements subscribe to data streams, manipulate the data and publish to the SVE. Agent elements subscribe to data within the network and provide data analysis and decision support capabilities. The modular structure of the RPMS enable the expansion, retraction, and reconfiguration capabilities to adopt to any analytical requirement. The dynamic data acquisition elements allow the system to adapt to environmental changes, e.g. changes in sensors types, data sources, etc. The network distribution capabilities enable collaboration of computation, allowing multiple analysis functions to be achieved through dedicated processing units. The resultant RPMS is decentralised in nature yet cooperatively united through asynchronous services.

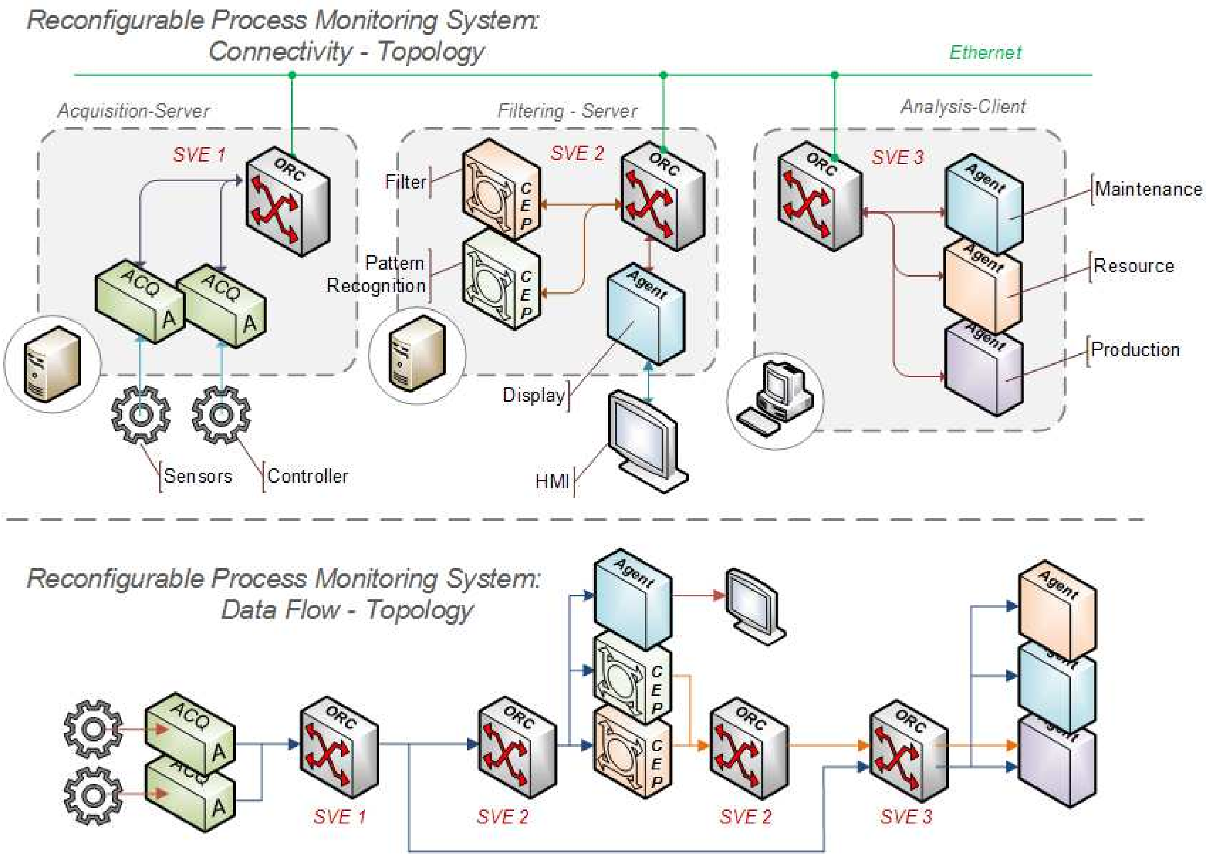


Fig. 7. Reconfigurable process monitoring system topology

3. CONCLUSION

Service oriented architecture provides the realisation of collaborative decentralised systems, which is the way for the future of cyber physical systems. The specific aim of this work was the introduction of a reconfigurable process monitoring system that utilises a service-oriented framework to achieve dynamic data acquisition and distribution in a manufacturing environment. The layered modelling approach enables a domain specific service model to be created through conceptual element characterisation, resulting in a comprehensive service mapping of data services to meet the needs of a reconfigurable process monitoring systems requirements. The results from the work identified the utilisation of binary conversion messaging methods to enable effective data structuring and compression for efficient data transmission. Results also identified the ability of the shared variable engine to provide interoperability within a local and networked environment. The combination of data servicing and message modelling resulted in a round trip message communication time of 0.88ms. This satisfies the 1ms criteria for real-time system requirements. This work has outlined the steps taken to characterise a decentralised system and enable an efficient and effective platform for implementation. The steps taken can be utilised to define other service orientated architecture systems by presenting a bottom up data model that can be abstracted to fit other requirements. Dynamic data acquisition and distribution capabilities within manufacturing systems are imperative due to the multi-domain and multi-functional requirements in manufacturing. Future work will focus on abstract analysis tools to utilise the reconfigurable and distributed nature of the architecture.

ACKNOWLEDGEMENT

This research was funded under the Graduate Research Education Programme in Engineering (GREP-Eng) which is a PRTLI Cycle 5 funded programme and is co-funded under the European Regional Development Fund.

REFERENCES

[1] MONOSTORI L., CSÁJI B.C., KÁDÁR B., PFEIFFER A., ILIE-ZUDOR E., KEMÉNY Z., SZATHMÁRI M.,. 2010, *Towards adaptive and digital manufacturing, Annu*. Rev. Control, 34/1, 118–128.

[2] ABELE E., LIEBECK T., WÖRN A., 2006, *Measuring flexibility in investment decisions for manufacturing systems*, CIRP Ann. - Manuf. Technol., 55/1, 433–436.

[3] ZAMBRANO REY G., PACH C., AISSANI N., BEKRAR A., BERGER T., TRENTESAUX D., 2013, *The control of myopic behavior in semi-heterarchical production systems: A holonic framework*, Eng. Appl. Artif. Intell., 26/2, 800–817.

[4] WESTKÄMPER E., 2007, Strategic development of factories under the influence of emergent technologies," *CIRP Ann. - Manuf. Technol.*, 56/1, 419–422.

[5] MORA D., TAISCH M., A. COLOMBO W., 2012, *Towards an energy management system of systems: An industrial case study*, IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society, 5811–5816.

[6] CASSINA J., ALESSANDRO C., TAISCH M., 2009, *Development of an extended product lifecycle management through service oriented architecture*, Industrial Product-Service Systems, CIRP IPS2, Cranfield University, 118–124.

[7] MENDES J.M., LEITÃO P.J., RESTIVO F.J., COLOMBO A.W., BEPPERLING A., 2008, *Engineering of service oriented automation systems: a survey*, Innovative Production Machines and Systems, Cardiff University, 33–38.

[8] JAMMES F., MENSCH A., SMIT H., 2007, *Service-oriented device communications using the devices profile for Web services*, in Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on, 1, 947–955.

[9] CANDIDO G., JAMMES F., de OLIVEIRA J.B., COLOMBO A.W., 2010, *SOA at device level in the industrial domain: Assessment of OPC UA and DPWS specifications*, Industrial Informatics (INDIN), 8th IEEE International Conference on. 598–603.

[10] LEITÃO P., MENDES J.M., BEPPERLING A., CACHAPA D., COLOMBO A., RESTIVO F., 2012, *Integration of virtual and real environments for engineering service-oriented manufacturing systems*, J. Intell. Manuf., 23/6, 2551–2563.

[11] JAMMES F., BONY B., NAPPEY P., COLOMBO A.W., DELSING J., ELIASSON J., KYUSAKOV R., KARNOUSKOS S., STLUKA P., TILL M., 2012, *Technologies for SOA-based distributed large scale process monitoring and control systems*, IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society, 5799–5804.

[12] KOREN Y., HEISEL U., JOVANE F., MORIWAKI T., PRITSCHOW G., ULSOY G., Van BRUSSEL H., 1999, *Reconfigurable manufacturing systems*, CIRP Ann. - Manuf. Technol., 48/2, 527–540.

[13] COLOMBO A.W., JAMMES F., SMIT H., HARRISON R., LASTRA J.L.M., DELAMER I.M., 2005, *Service-oriented architectures for collaborative automation, in Industrial Electronics Society, 2005, IECON 2005. 31st Annual Conference of IEEE, 2649–2659.*

[14] NAGORNY K., COLOMBO A.W., FREIJO E., DELSING J., 2013, *An engineering approach for industrial SoA-based systems of systems*, Industrial Technology (ICIT), 2013 IEEE International Conference on, 1956–1961.

[15] BONY B., HARNISCHFEGER M., JAMMES F., 2011, *Convergence of OPC UA and DPWS with a cross-*domain *data mode*l, in Industrial Informatics (INDIN), 9th IEEE International Conference on, 187–192.

[16] TETI R., JEMIELNIAK K., O'DONNELL G., DORNFELD D., 2010, *Advanced monitoring of machining operations*, CIRP Ann. - Manuf. Technol., 59, 717–739.

[17] KARNOUSKOS S., COLOMBO A.W., BANGEMANN T., MANNINEN K,. CAMP R., TILLY M., STLUKA P., JAMMES F., DELSING J., ELIASSON J., 2012, *A SOA-based architecture for empowering future collaborative cloud based industrial automation*, in 38th Annual Conference of the IEEE Industrial Electronics Society (IECON), Montréal, Canada, 5770–5776.

[18] PIETRZAK P., LINDGREN P., MAKITAAVOLA H., 2012, *Towards a lightweight CEP engine for embedded systems*, IECON - 38th Annual Conference on IEEE Industrial Electronics Society. 5805–5810.

[19] DELSING J., ROSENQVIST F., CARLSSON O., COLOMBO A. W., BANGEMANN T., 2012, Migration of industrial process control systems into service oriented architecture, *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*, 5786–5792.

[20] CÂNDIDO G., BARATA J., COLOMBO A.W., JAMMES F., 2009, *SOA in reconfigurable supply chains: A research roadmap*, Eng. Appl. Artif. Intell., 22/6, 939–949.

[21] SAVIO D., KARNOUSKOS S., WUWER D., BANGEMANN T., 2008, *Dynamically optimized production planning using Cross-Layer SOA*, Computer Software and Applications, COMPSAC, 32nd Annual IEEE International, 1361–1365.

[22] SPIESS P., KARNOUSKOS S., GUINARD D., SAVIO D., BAECKER O., SOUZA L., TRIFA V., 2009, *SOA-Based integration of the Internet of things in enterprise services*, Web Services,. ICWS, IEEE International Conference on, 968–975.

[23] GUINARD D., TRIFA V., SPIESS P., DOBER B., KARNOUSKOS S., 2009, *Discovery and on-demand provisioning of Real-World Web Services*, Web Services, ICWS, IEEE International Conference on, 583–590,

[24] KYUSAKOV R., ELIASSON J., DELSING J., 2011, *Efficient structured data processing for web service enabled shop floor devices*, Industrial Electronics (ISIE), IEEE International Symposium on, 1716–1721.

[25] JAMMES F., 2011, *Real time device level Service-Oriented Architectures, Industrial Electronics (ISIE),* IEEE International Symposium on, 1722–1726.

[26] KYUSAKOV R., MAKITAAVOLA H., DELSING J., ELIASSON J., 2011, *Efficient XML interchange in factory automation systems*, IECON - 37th Annual Conference on IEEE Industrial Electronics Society, 4478–4483.

[27] National-Instruments, "Network Variable Technical Overview", 2009.

[28] National-Instruments, "Using the LabVIEW Shared Variable", 2012.

[29] KARNOUSKOS S., SOMLEV V., 2013, *Performance assessment of integration in the cloud of things via web services*, Industrial Technology (ICIT), IEEE International Conference on, 1988–1993.