Vertical integration in industrial enterprises and distributed middleware

Ahmed Ismail* and Wolfgang Kastner

Institute of Computer Aided Automation,
Vienna University of Technology,
Vienna, Austria
Email: aismail@auto.tuwien.ac.at
Email: k@auto.tuwien.ac.at
*Corresponding author

Abstract: Recent advances in information, operations and networking technologies have brought forth many economical and organisational opportunities for industrial enterprises. One of the enabling pillars for the exploitation of these opportunities hinges predominantly on the design and introduction of frictionless vertical integration in enterprises. The paper presented concerns itself with the generation of autonomous middleware for vertical integration in existing or new industrial enterprises. Consequently, the paper discusses the main concerns of designing such middleware systems for typical enterprise infrastructure. Solutions from both the information and operations technology sectors are examined in light of said requirements with the aim of enhancing the current state of middleware solutions and industrial enterprises alike.

Keywords: middleware; vertical integration; industrial internet of things; industrial enterprise; enterprise reference architecture; ERA; industrial network design; factory connectivity standards; peer-to-peer; P2P; virtualisation; IEC 61499; middleware architectures.

Reference to this paper should be made as follows: Ismail, A. and Kastner, W. (2016) 'Vertical integration in industrial enterprises and distributed middleware', *Int. J. Internet Protocol Technology*, Vol. 9, Nos. 2/3, pp.79–89.

Biographical notes: Ahmed Ismail is a PhD candidate at the Automation Systems Group of TU Vienna. Since joining TU Vienna, he has been responsible for researching scalable and secure solutions related to the future of factory communication infrastructures. His work pertains to various system integration techniques for industrial automation systems.

Wolfgang Kastner is an Associate Professor at TU Viennas Faculty of Informatics. His research interests concern networked control systems and distributed automation systems with a special focus on system integration (enterprise integration; field level integration) taking into account safety and security issues. His areas of interest include cyber-physical production systems, smart buildings and smart grids, and their seamless integration into the internet of things.

1 Introduction

As noted in the Reference Architecture Model Industrie 4.0 (RAMI 4.0), a principle element in the design of modern industrial systems for enhanced enterprise competitiveness is vertical integration (Adolphs et al., 2015). The importance of vertical integration is derived from the inherent complexities typical of distributed automation systems. That is, industrial automation systems currently utilise a large number of protocols and are present in many various architectural forms. This heterogeneity proves to be a major hindrance to the adoption of modern technologies that are increasingly dependent on the easy accessibility of data and devices. Vertical integration is primarily concerned with addressing this heterogeneity by enhancing interlayer data exchange capabilities, thereby allowing for the deployment of contemporary business-enhancing applications. A typical solution employed in the pursuit of vertical integration is the use of gateways capable of protocol transformation. Although a viable strategy for use in the industrial IoT (IIoT), transformation may result in information loss and is a costly engineering effort to develop and integrate with existing systems.

Nevertheless, extensions to the basic concept of gateways warrant their complex and costly engineering effort. By increasing the level of integration between automation layers and full enterprises you allow for the integration of meaningful services. That is, the gateway may be elevated through the pervasive deployment of the gateway hardware and the introduction of enhancements such as web portals and data-enriching information models. In doing so, the gateway achieves an additional layer of intelligence. This intelligence, afforded the property of omnipresence by the middleware, may host a myriad of services appropriate for any industrial layer. The closest of analogies to such an envisioned system would be the natural

evolution of the enterprise service bus (ESB) towards an industrial gateway service bus (GSB). Such a middleware system may thereby add a level of robustness, versatility and extensibility to the industrial enterprise, providing it with the ability to integrate a wide array of modern algorithms and technologies in a safe and secure manner. The importance of such technologies lies in the fact that they may then be used to enrich the process. It could, for example, allow for an increase in a plant's efficiency and be used to facilitate a plant's compliance with legal and regulatory standards, or even assist in enhancing its energy efficiency or infrastructural security.

We concern ourselves in this paper with a specific type of middleware; that being an autonomous middleware capable of being deployed on dedicated or regular industrial hardware, such as field level elements, for the purposes of vertical integration in existing or new enterprises. Consequently, the coming sections address several topics that need to be considered when designing such a middleware. Sections 2 and 3 present overviews on the concepts of enterprise reference architectures (ERA) and ERA-derived network designs, respectively, along with discussions on methods of middleware integration with such ERAs and networks. Section 4 summarises the various factory connectivity standards that the middleware may use to interface with an industrial system. Section 5 provides a description of two prominent projects that use connectivity standards to generate holistic SOA middleware solutions. Section 6 presents a description of several solutions from the information technology (IT) and operations technology (OT) domains that may be used to extend the capabilities of middleware systems. Section 7 provides an information-centric viewpoint of middleware and process security. Finally, the paper concludes in Section 8.

2 Industrial ERAs

A reference architecture is a structured meta-model representing the various functional elements and interactions of an enterprise system. Industrial enterprises use a reference architecture in order to allow for the rapid generation of a useful system architecture that adopts all of the relevant insights and best practices gained from years of previous deployments. Developing middleware without taking into account the restrictions imposed by existing architectures risks the breaking of application dependencies vital to the middleware once it is deployed. In this section, we will focus predominantly on the Purdue enterprise reference architecture (PERA) framework as it is widely accepted by industry and is compatible with multiple standards, such as ISA 95, ISA 88, and IEC 62443 (Giachetti, 2011; Williams, 1993; He et al., 2012).

Using the ISA 95 and ISA 88 models, PERA essentially segregates the elements comprising an industrial enterprise system into separate zones and conduits. The system is essentially divided into five functional layers. The lowest layer, level 0, is the actual physical process. Level 1 consists of the device communication networks directly in control of the physical processes. The second layer comprises the control and automation network. This level can directly access the process and discrete devices of level 1 to set or reconfigure them as needed. Level 3 consists of operations management systems, such as the MES. Level 3 components may only read from layers 1 and 2. Level 4 is where the enterprise system is located and is where the bulk of the business process network exists. The application of the PERA model has been extended with time and currently includes a sixth layer and a DMZ. The sixth layer, level 5, is where centralised IT systems and their associated functions are situated. The DMZ, on the other hand, is in place to manage access from levels 4 and 5 to the data and network of levels 0–3. The recommended practice is to have no traffic cross the DMZ, instead all traffic should either originate or terminate within the DMZ. Therefore, data sharing and application servers are normally found in the DMZ (Williams, 1993; He et al., 2012; Didier et al., 2011).

Beyond segregation based on function, an enterprise's architecture is also influenced by a number of other aspects. Discussed in Zerbst et al. (2013), and reproduced in Table 1, these include elements such as security assessments and operations and maintenance work. Such factors imply that the enterprise infrastructure and architecture is dynamic in nature as it must adapt opportunistically in line with their results over time. In order to allow for a truly autonomous middleware, the middleware system must therefore have methods through which it is able to continuously and accurately infer the existing infrastructure within which it is operating.

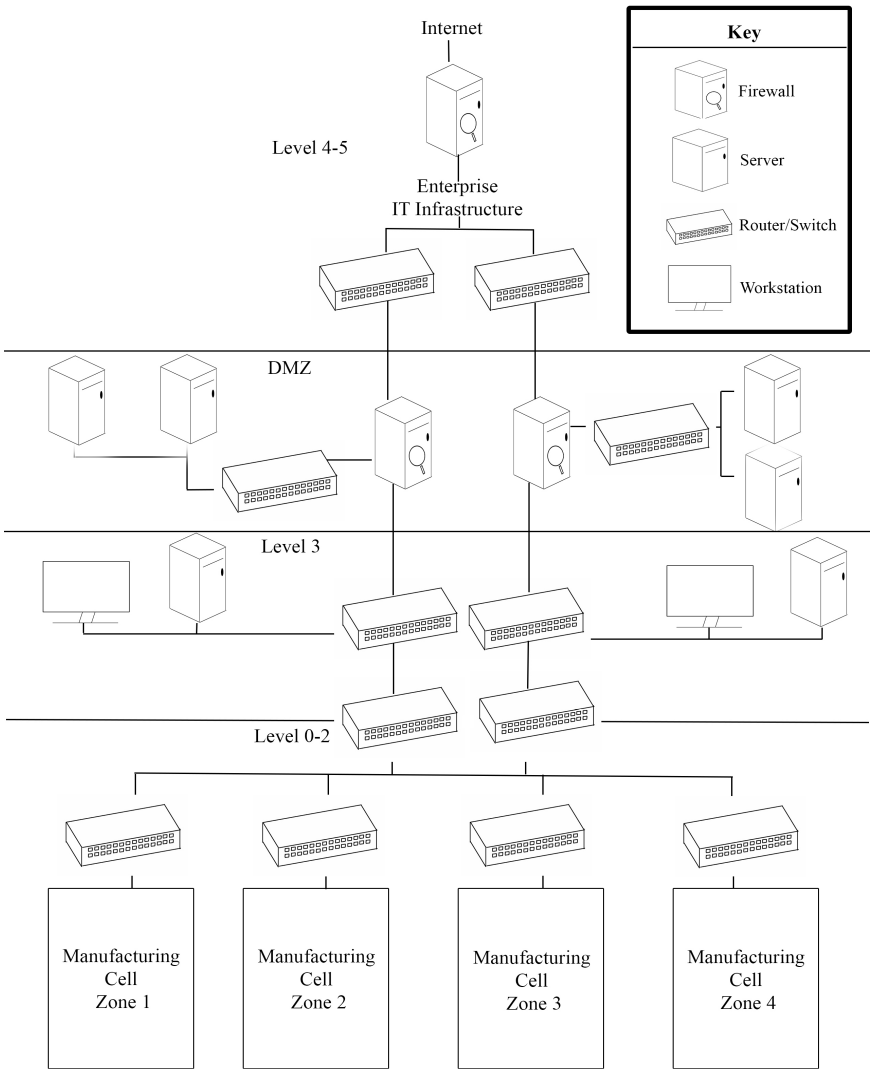
As noted in Hauder et al. (2012) and Farwick et al. (2013), literature exploring automated methods of acquiring such information is quite limited. The work of Hauder et al. (2012) lists the challenges facing automated EA modelling. These include having to deal with large volumes of data, generating the appropriate models to transform data into useful representations, and using the tools to do so in an appropriate and secure manner. The GSB domain typically deals with the secure acquisition, processing, transformation and extraction of meaningful information from data (Colombo et al., 2014). Consequently, this places the GSB at an ideal starting point for tackling problems of integration with dynamic environments.

Table 1 Governing factors in zonal population

Factor	*Explanation*
Regulations and legislation	Systems must observe the imposed local, regional, national or international regulations, legislation and standards.
Impacts	The impact of system malfunctions on the business must reflect on the system design.
Safety	The system must comply with safety best practices. Standards such as IEC 61508 and IEC 61511, provide the necessary guidance.
Security	The system must be designed in accordance with the requirements of security risk assessments.
Locality	The physical location of systems must be taken into consideration when defining both the zones and the digital system in itself.
Architecture	The system must integrate within the overall technical and landscape architecture.
Operations and maintenance	Operational and maintenance considerations impacting the system, the technical architecture, or the physical infrastructure, should be taken into account.
Organisation	The organisational structure and culture introduces a set of requirements that ought to be reflected unto the system under design.

Source: Zerbst et al. (2013)

Figure 1 A typical ethernet-based industrial enterprise network



Source: Didier et al. (2011)

The main concern for integration is therefore in the selection of the appropriate tools for the acquisition of the necessary information. This is addressed in Farwick et al. (2013) where a survey shows that IT-oriented tools, such as network monitors and scanners (NMS), configuration management databases (CMDB), and ESBs are fully-automated solutions that provide the most actual data. Since, the GSB is based on the concept of ESBs, furthering the reach of the GSB would require either the provision of appropriate interfaces for these tools, or having said tools directly integrated as components of the GSB.

3 Network designs and considerations

Abstracting the reference architecture on the network design, Figure 1 shows a typical implementation of an industrial enterprise network. The way that such a network would be organised is as follows. Any traffic entering the enterprise from the internet is intercepted by a fifth layer DMZ. Alternatively, a firewall may be used to isolate or inspect and filter packets entering the enterprise. To access the lower layers from the enterprise a second DMZ must be crossed. Figure 1 shows two firewalls and two routers controlling access to the DMZ, this is only to highlight the manner in which redundancy requirements may reflect on the network design. This element of redundancy is continued throughout the layers with the left and right portions of Figure 1 representing redundant versions of the same enterprise components. Traffic into the DMZ, whether from the enterprise or from a lower layer, traverses a firewall before accessing the required servers. This is to filter and prevent the direct communication of levels 4–5 and levels 0-3. Level 3 and below consists of process and control servers which would be accessed using a myriad of switches or routers. These switches and routers separate the various areas of concern into VLANs or subnets, as required. These lower layers may contain industrial ethernet solutions or wireless technologies such as the ISA 100.11a, Wireless HART, or OCARI standards, which, in turn, would require gateway solutions. Finally, technologies such as NAT may also be present at the lower layers to simplify address management during the reconfiguration of manufacturing cells (Didier et al., 2011).

Integrating the middleware with the network brings forth two concerns; firstly, the extraction of data without impacting communication links, and secondly, allowing for middleware components to communicate across layers without requiring extensive changes to the infrastructure.

The former of these two points has to do with the heterogeneous nature of the enterprise layers. Due to the varying application needs and constraints of each of these layers, the protocols that operate within each respective layer differ from one another. For the lower layers, for example, real time connectivity is a necessity, while for the upper layers, it is normally not a requirement. In order to allow for effective protocol convergence and cross-domain communication, the middleware needs to be able to intercept all possible communication paths to and from levels 0-DMZ. This needs to be done without impacting the constraints or design factors binding said paths. Fortunately, a number of methods exist that may allow the middleware to do so. These include port mirroring, and more intrusive methods such as network TAPs and bridging (Didier et al., 2011; NSA, 2010).

However, in regards to cross-layer communication by the middleware in the face of NAT, the inclusion of NAT traversal may be required. NAT traversal techniques depend on the mode of communication and either involve UDP hole punching or TCP hole punching. UDP hole punching may use STUN or TURN. TCP hole punching operates similarly to its UDP counterpart and includes mechanisms such as ATUNT, NUTSS and NATBlaster (Phuoc et al., 2008).

4 Factory connectivity standards

The factory connectivity standards to be discussed here are a collection of technologies used to define standardised interfaces and methods for information presentation and description. This standardisation allows for vendor-neutral interoperability and efficient information exchange between the various functional elements of an industrial enterprise. In addition to allowing the middleware to integrate with the enterprise, they may also be used in intra-middleware communication. The section addresses a subset of both information-centric and message-oriented middleware concepts. The information-centric standards will provide a description of holistic solutions, information modelling and schema specifications, a solution for web services integration, and a standard for compact XML data exchange for constrained devices. On the other hand, the message-oriented middleware subsection will focus on real-time communication standards that lack information models and schema.

4.1 Information-centric standards

Object linking and embedding (OLE) for process control (OPC) classic is a client-server-based set of standards for the definition of data transfer software interfaces. This grouping of complementary standards consists of the data access (OPC DA), historical data access (OPC HDA), alarms and events (OPC A&E), data exchange (OPC DX), security, XML-data access (OPC XML-DA), complex data, commands, batch, and OPC .NET specifications. By standardising these interfaces, a common method is available for the exchange of data between diverse control systems and products. This greatly simplifies the driver-development process for the vendors and manufacturers of industrial systems. The data exchange process itself uses Microsoft's distributed component object model (DCOM) (Byres et al., 2007).

The OPC unified architecture (UA) standard is the successor of OPC classic. It addresses some of the shortfalls of OPC classic, extending it appropriately in order to allow for platform independence, object orientation and a service

oriented architecture (SOA), all while maintaining backwards compatibility with OPC Classic. Functionally, the OPC UA information model is a combination of OPC DA, A&E, HDA, Commands, and Complex Data, and is built on OPC DA's data model. The services defined by OPC UA consist of a fixed set of groups; namely, they are the Discovery, SecureChannel, Session, NodeManagement, View, Query, Attribute, Method, Subscription, and MonitoredItem services. To ensure interoperability, these sets are fixed in behaviour and parameters. For security, the features included are able to provide for authentication, authorisation, confidentiality, integrity, freshness, auditability, and availability. Reference implementations are available in a variety of languages, including ANSI C, Java, and .NET, and as such, may be deployed on systems from the embedded to the enterprise level. As of yet, OPC UA is not able to operate under real-time constraints; however, the OPC Foundation is currently exploring the use of IEEE 802.1 TSN to allow for such real-time capabilities (Mahnke et al., 2009; B&R, 2015).

Automation Markup Language (AutomationML) is an XML-based neutral data format designed specifically for the storage and transfer of plant engineering information. AutomationML is aimed at connecting together the tools of the various engineering disciplines and phases of the plant life cycle. It does so by having all of the modelled and stored information follow a neutralised hierarchical object-oriented model. That is, the various engineering tools being incorporated in the information model have their object identification schemes homogenised by a single model. This model is governed by a set of three existing standards, along with a defined set of rules on how these standards may be used and interlinked. Together, these standards form the AutomationML specification. In detail, AutomationML uses the CAEX standard (IEC 62424) for plant topology information, COLLADA for geometry and kinematics information, and PLCopenXML for the modelling of control, behaviour, sequencing and logics information (AutomationML, 2014; Henßen and Schleipen, 2014).

The MTConnect standard is a four part XML-based standard for data integration. It provides a read only, RESTful data retrieval mechanism over HTTP, thereby uniformly supplementing existing systems' transport mechanisms regardless of their heterogeneity. The XML schema defined by MTConnect defines information models for devices, streams, and assets. MTConnect defines devices as functional machines or parts of machines. Streams are mostly concerned with organising devices, events and samples. Finally, assets are physical objects that are not detrimental for the functionality of machinery. Extensions to the standard exist that define models for device interfaces and for cutting tools as assets (Sobel, 2015).

The device profile for web services (DPWS) OASIS standard is an architecture used to provide an enterprise with a host of web services (WS) technologies. It also uses WS-protocols in order to allow for seamless peer-to-peer (P2P) device interactions. Architecturally, DPWS is composed of hosted and hosting services. The hosted services provide functionality to a device, while the hosting services provide the hosted services with discovery capabilities. In all, the DPWS built-in services include discovery, metadata exchange, and publish/subscribe services. However, DPWS may be extended using the appropriate WS-specifications to allow for security, resource and device management, eventing, addressing, policy management features, as well as a number of other functionalities (Hock et al., 2009; Driscoll and Mensch, 2009; Sucic et al., 2012).

4.2 Message-oriented middleware

Several existing implementations of MOMs include, but are not limited to, DDS, CORBA, MQTT, AMQT, STOMP, and ZeroMQ. However, the focus of this section will be on industrially-proven real-time (RT) communications standards. Consequently, this section only covers the DDS and CORBA standards.

DDS is a data-centric OMG middleware standard capable of both dynamic discovery and of implementing QoS parameters in order to provide its publish/subscribe mechanisms with real-time (RT) communications. Its ability to support RT performance has allowed for its deployment in mission critical systems (Schmidt and Hag, 2008). The DDS interoperability protocol is defined by the real-time publish subscribe (RTPS) DDS specification (IEC-PAS-62030). The RTPS wire protocol allows devices from multiple vendors to communicate by multicasting IP-based connectionless best-effort transport protocols such as UDP (OMG, 2014). Implementations of the DDS standard exist in both open source form, such as is the case with OpenDDS and OpenSplice, and in commercial forms, such as RTI-DDS. The OMG's middleware and related services (MARS) platform task force (PTF) has, as of recently, set itself a target for defining a gateway in order to bridge together the DDS and OPC UA standards (Ungurean et al., 2014; Wales, 2015; RTI, 2009).

As opposed to DDS, the common object request broker architecture (CORBA) is a client/server OMG middleware standard used for the transfer of information between heterogeneous applications and systems. For real-time guarantees the RT-CORBA specification may be used. RT-CORBA allows for deterministic access to shared resources and has several scheduling policies in place for multithreading applications. Like DDS, CORBA is platform independent and type safe. However, unlike DDS, CORBA is a point-to-point standard that is not capable of dynamic discovery; rather, nodes must know each other directly or use a naming service for discovery. Furthermore, CORBA lacks the QoS abilities of DDS that allow for the tailoring of communications. Lastly, the interoperability wire protocol specified by CORBA is part of the CORBA standard itself, compared to DDS which is independent of a wire protocol and only has one defined as a non-mandatory accompanying specification. Currently, as pointed out in Ungurean et al. (2014), several open-source and commercial

implementations of CORBA exist, with the most prominent open-source implementation being The Ace ORB (TAO) project (Ungurean et al., 2014; OMG, 2005).

As is apparent from the descriptions of the standards in the previous two subsections, a number of these standards take on many of the different aspects required to enrich the factory process. These include functions related to integrating web services, RESTful data retrieval, and the integration of information exchange models that guarantee access to enriched field level data. However, since constraints and requirements vary from one level of the enterprise to the other, none of the standards above are capable of providing these features for the entire enterprise. In order to address such shortcomings, middleware architectures have been presented in recent years merging together some of the aforementioned standards, and extending them appropriately in order to create complete architectures and communication stacks. The coming section will discuss two of these architectures that are designed specifically for vertical and horisontal integration.

5 Middleware architectures

The architecture and communication stack governing the middleware may take on several forms in order to ensure interoperability and efficiency of communication amongst the middleware components. Within this section, a review of two large EU FP7 project outcomes is presented in order to give an overview of the different considerations and strategies put to use in the development of such an architecture and stack. These are the architecture for service-oriented process – monitoring and control (IMC-AESOP) and IoTSys projects. The former is a middleware solution that presents a SOA for industrial plants. IoTSys, on the other hand, is a project that uses a communication stack and a middleware solution to provide smart IoT objects with interoperability and plug-and-play connectivity in the building automation domain (Colombo et al., 2014; Jung, 2014).

5.1 IMC-AESOP

The architecture for service-oriented process-monitoring and control (IMC-AESOP) project investigates the use of a SOA to achieve monitoring and control in extensively large process control systems. The goals addressed by this project include, amongst many others, the management of interoperability, plug-and-play characteristics, and self-x properties. In order to achieve such goals, the project presents a large set of clearly defined interdependent service groups. These service groups are generated based on a number of well-defined use cases. These use cases include design considerations such as asset monitoring, backward/forward compatibility, cross-layer integration, and others that are considered to be of key importance to industrial enterprises. Based on these considerations, a large set of clearly defined interdependent service groups are proposed. Access to these services and their data is governed by access controls outlined for clearly set user roles. Technically, the design of the SOA-compliant prototype considers the use of a number of technologies that would allow for a web service compliant SOA to span across the entirety of the ISA-95 enterprise architecture (Colombo et al., 2014).

Technologically, the system presents a wide array of contributions. One of these is the bridging of OPC UA and an extended form of DPWS that incorporates the EXI specification and semantic data models. In doing so it creates a hybridised web services profile enriched with a data model that can provide for all of the layers of the enterprise while having extensively resilient and scalable abilities for plug-and-play discovery. The system follows the distributed paradigm and is deployable on resource constrained components, such as embedded systems, and on resourceful ones, where more extensive features would be present. Furthermore, the system presents and implements a mediating gateway which, using legacy interfaces and semantic data models, supports plant migrations from legacy to SOA architectures by allowing the two systems to communicate throughout the migration process. One of the main features of the system is the complex event processing toolset which uses alarm load-shedding and state-based alarming to resolve the issue of alarm flooding. The architecture developed was successfully deployed in a number of settings, including plant lubrication monitoring, plant energy management, and district heating and energy management. These use cases demonstrate the viability of the developed system (Colombo et al., 2014).

5.2 IoTSys

As part of the IoT6 project, IoTSys presents a middleware for the integration of heterogeneous and legacy devices for the internet of things. Although the system is focused on the IT domain of building automation (BA) systems, its relevance is in its design. This design demonstrates an architecture that provides a holistic and concise solution to managing heterogeneity in a similarly complex field (Jung, 2014).

The IoTSys design proposes a complete communication stack for IoT-devices centred around the use of the IPv6 protocol. Therefore, the networking layer of the stack uses the IPv6 protocol and is secured using the IPsec protocol suite. This allows the lowest layers to integrate smart objects using any IPv6-compatible protocol. This includes protocols such as 6LoWPAN, the IEEE 802.15.4-based standards, Bluetooth low energy (BLE), IEEE 802.11, and IEEE 802.3 ethernet (Jung, 2014).

For message exchange, the stack also proposes the use of HTTP, CoAP or SOAP for reliable TCP-based, unreliable UDP-based, RESTful or other forms of communication. This allows the stack to interact with both constrained and resourceful devices. Based on these protocols, the exchange may be secured using DTLS/SSL or WS-Security, appropriately. The payloads themselves may be encoded using XML, JSON, EXI or oBIX's binary encoding, and secured using XML Encryption and XML

Signature. This, again, allows the system a degree of flexibility come time of integration with the various building automation applications that may be presented to it (Jung, 2014; Kastner et al., 2014).

As for the services themselves, the stack addresses device and service discovery, service description, DA, eventing, HDA, group communication, and authorisation. Device discovery is done using multicast DNS (mDNS). The services, on the other hand, use DNS service discovery (DNS-SD) for service exposure and IoT-oBIX contracts for service description. For authorisation purposes, the stack uses the XACML architecture to define and process access requests to devices and resources. Finally, all of the remaining services are defined using the oBIX specification (Jung, 2014; Kastner et al., 2014).

6 Integrating IT and OT domain solutions

The existing requirements for designing a system for vertical integration have, as has been shown in the previous section, been tackled with a heavy focus on the use of factory or building automation connectivity standards. That is, there has been an almost exclusive focus on protocol convergence techniques. In contrast, this section proposes that, in addition to the typical approach of utilising connectivity standards, further solutions from the IT sector should be integrated with OT in order to enhance middleware solutions. Consequently, this section discusses the integration of two IT domain solutions, virtualisation and P2P networking, and one OT solution, IEC 61499, to address the abstract set of requirements summarised in Table 2.

The first of these technologies, virtualisation, is to introduce language-neutral OSGi-like functionality and security features through compartmentalisation. P2P networking, on the other hand, is to introduce aspects of data and service idempotence to the system in order to allow for survivable data and infrastructure. Finally, IEC 61499 will be discussed as a form of standardising the structure and interface of middleware components to ensure portability, interoperability, and a number of other features that are necessary for distributed systems. It is important to note that due to the many facets involved in designing a secure system, aside from virtualisation, the solutions for the security requirement will be addressed separately in Section 7.

6.1 Virtualisation

One of the advantages of virtualisation is its inherent ability to allow for a higher level of security by isolating separate workloads from each other (Christodorescu et al., 2009). If the executing service components of a SOA are designed in a modular fashion, then they may be deployed as stand-alone applications dispersed across a number of virtual machines. To communicate, these applications may interact with each other using a predefined communication stack. Architected properly, the system will stand a better chance of containing the damage caused by a single compromised service.

The modular and isolated fashion of the services also simplifies the testing and commissioning of new versions of binaries. This is since the use of virtualisation allows for the inheritance of OSGi-like capabilities of being able to dynamically apply start, stop, pause, and other operations on the VM, and, consequently, on the running service (Redondo et al., 2008). Therefore, separated through virtualisation, two versions of the same service may be deployed at the same point in time and on the same device in the industrial plant. If the new version of the service is found to be unstable, it may be paused or halted while the stable binary continues to offer the service uninterrupted. Alternatively, once the newer version is determined safe for consumption, then, again, the older version may be halted without any interruption to the availability of the service being offered. This allows for a tolerant form of version commissioning in a safety-critical environment.

Table 2 Requirements for vertical integration and proposed solutions

Requirement	*Components*	*Solutions*
Fidelity to ERA	Information extraction	NMS, CMDB, GSB, port mirroring, IGMP snooping, network TAPs, and bridging
	Distributed partial deployments of the SOA	Connectivity standards, virtualisation, P2P networking, and IEC 61499
Cross-layer communication	Accommodation of firewalls, VLANs, and NAT	NAT traversal techniques and P2P networking
Interoperability	Interfacing and module standardisation	Connectivity standards and IEC 61499
Scalability	Plug-and-play and self-organisation	Connectivity standards and P2P networking
Fault tolerance	Reliability, availability, and serviceability	Virtualisation and P2P networking
Functional safety	Timing and determinism in mixed criticality systems	Connectivity standards and virtualisation
Security	Access controls, authorisation, identification and authentication, incident response, communication protection, information integrity	Virtualisation, encryption, tunnelling, and data lifecycle mechanisms

Source: Wilamowski and Irwin (2011) and Radvanovsky (2013)

Currently, embedded virtualisation solutions exist atop which containers, full VMs, or unikernels may be virtualised. VM and container virtualisation technologies are both mature solutions that have many solutionspecific performance comparisons in existence to guide decisions. An interesting new virtualisation technology is that of unikernels, such as Rump Kernels, MirageOS, and ClickOS. These are single-purpose machine images compiled from application code, configuration files, and libraries into stand-alone kernels (Madhavapeddy et al., 2013). As such, in comparison to containers and VMs, they tend to have very small image sizes, tiny memory footprints, small attack surfaces for exploitation, and are very quick to boot (Madhavapeddy et al., 2013). Such features give them an obvious advantage over containers and VMs for a variety of applications, including industrial middleware. However, at this point in time, unikernels are still at an early stage of development and, as such, have been targeted specifically at web development and network function virtualisation (NFV) applications. The developments required to allow for their deployment in industrial settings is still far from realisation. This limits the current viable options to containers and VMs.

Due to current advances in the virtualisation sector, we expect to see solutions emerge from this domain for time-critical industrial control applications as well. This is since there exists no technical reason that would prevent embedded systems from hosting RTOS guest VMs. Such a merger between these two technologies has already been proposed and explored for time-critical applications in the automotive, mobile communications, industrial, and other domains (Gu and Zhao, 2012; Bregenzer and Adämmer, 2010). Consequently, several solutions for the para-virtualisation and full virtualisation of RTOS systems, such as Seehwan Yoo and Yoo (2014) and Avanzini et al. (2015), respectively, have already been proposed and validated experimentally. The opportunity therefore exists to build upon the same technologies in order to allow for any single element of a distributed middleware system to dynamically provide for the very nature of mixed criticality that is typical of industrial enterprises.

Finally, with respect to fault tolerance, virtualisation brings forth aspects that are applicable to the concepts of redundancy for availability within the context of industrial domains. Simply, if a fault manifests itself as an error on a device, threatening the functionality of operational services, the concepts of VM migration ensure that a replica of the running VM may be transferred to a functional device. Normal migrations requires the VM to be suspended for the entire time that the memory is recorded before transfer. Alternatively, live migration transfers differential snapshots causing the VM to be suspended for substantially shorter periods of time while the transfer occurs. This allows the system to appear as though it has suffered no interruptions. Such a mechanism is not only useful for fault tolerance, but also in the case of ensuring continued availability during any process that requires the device to be taken offline, such as invasive maintenance procedures (Ando et al., 2009).

6.2 P2P networking

Infrastructure idempotence may be enhanced by merging virtualisation with P2P networking technologies. P2P solutions may allow for a cohesive, fault-tolerant network of middleware components for the non-real-time management, replication, storage and sharing of plant data and service components.

Currently, P2P technologies may be classified as centralised and decentralised networks, structured and unstructured networks, hybridised (partially decentralised) networks, horizontal hierarchical (HoHA), vertical hierarchical (VeHA), and hybrid hierarchical (HyHA) networks. Of these, VeHA architectures appear to be the most versatile implementation. This is since VeHA networks divide their member nodes into layers, creating a DHT for each layer, and then requiring the use of gateways, such as co-located nodes, for inter-layer communication. This means that VeHA systems are capable of tolerating and operating in the presence of NAT and firewalls. An added advantage of VeHA networks is their ability to properly interact with a heterogeneous middleware infrastructure made up of components with varying resources (Ou, 2010).

Using VeHA networks, the topic of P2P networks as systems may find an application domain within the industrial environment. The definition of P2P networks as systems is a method for enhancing cross-layer functionality by bridging or merging together heterogeneous and homogeneous P2P networks. This allows for expanded systems, inter-system content-sharing, and inter-system traffic engineering. As systems the various components of the middleware may therefore be able to perform cross-layer resource-sharing tasks, to execute a single operation cooperatively, or simply to engineer more efficient paths for communication using the existing network infrastructure. This increases the survivability of the middleware infrastructure and enhances the middleware's capabilities at networking and at performing resource-intensive tasks (Ngo, 2013).

6.3 IEC 61499 function blocks

To standardise the system components and interfaces, and to push the same principles of design of modularity in the middleware down to the shop floor, the IEC 61499 standard may be useful. The IEC 61499 standard defines a distributed architecture for software development in the industrial control and automation domain. It is designed to allow for software portability, interoperability, reconfigurability, and encapsulation. This standard requires that software be implemented as a series of function blocks (FB), of which there are three classes; namely, the basic, composite and service interface FBs. Each FB encapsulates a processing algorithm and has explicit definitions of its required event and data inputs and outputs. For networking purposes, the standard also defines a communication interface FB (CIFB) that allows for both UDP publish/subscribe and TCP client/server modes of communication; however, it does not directly interfere with

the mechanisms of the networking layer. Using these function blocks, it is therefore possible to generate complex networks of interconnected FBs thereby creating an organised and well-coordinated distributed control system. As has been pointed out in Basile et al. (2013) and Dai et al. (2015), such mechanisms make IEC 61499 perfectly compatible with SOA. This is as the event-based IEC 61499 standard requires the FBs to have well-defined interfaces, thereby allowing separate FBs to be viewed as independent services ready to be consumed by external requesters whom only need to know the structure of the interface. This concept is built upon even further in Dai et al. (2015), where the SOA is used to allow for the dynamic reconfiguration of FBs. All in all, the principles of the IEC 61499 standard may be used to design and standardise all of the services offered by a middleware system, providing a SOA with extensive features that simplify its deployment on variably-resourced devices in a heterogeneous environment (Thramboulidis, 2013; Basile et al., 2013; Wilamowski and Irwin, 2011).

7 Security

The final topic of discussion in this paper is that of security. Security in the industrial domain has traditionally followed the approach of isolated islands (Wilamowski and Irwin, 2011). However, the increasing connectivity brought about through vertical integration necessitates the use of different strategies. The deployment of a pervasive middleware infrastructure presents a unique opportunity to introduce the required novel strategies through re-design and the elimination of fragility from the system, albeit, at the cost of requiring protection measures to secure the middleware devices themselves, and their data. The topic of discussion of this section will be limited to an information-centric viewpoint of securing the industrial process. This translates to protecting information from the point of creation to the point of destruction.

At creation, whether coming from or proceeding to a field level device, data transmission normally occurs using a fieldbus technology. Unadulterated, these protocols mostly implement either no or limited security measures; for example, Foundation Fieldbus, WorldFIP, Interbus and ControlNet all use unencrypted passwords for authentication. Therefore, to secure fieldbus protocols, the tunnelling of encrypted packets using fieldbus technology becomes a necessity. Unfortunately, fieldbus packets are very limited in size. This in turn severely limits the security enhancements that may be introduced. Furthermore, a tunnelling-based method requires that all end-devices be modified with the appropriate interfaces and application-level components. Since the replacement of all manufacturing equipment is not always a feasible option, this necessitates the implementation of varying security profiles which are able to provide different levels of guarantees based on context (Wilamowski and Irwin, 2011).

In addition to influencing the confidentiality of data, context also influences the management of data. That is, factors of context must also be taken into consideration when assigning proper access rights to data before storage. This is in order to ensure that any future use of the data is only possible by the correct machine or user and only under the proper circumstances. Such rights may be assigned at the point of creation by capable field devices, or, as would be the case for legacy devices, by GSB devices that would intercept and augment non-compliant data. At the point of storage, XACML, enterprise digital rights management (EDRM), and structured data tagging are suitable access management solutions that may be applied to structured or unstructured data. In addition to being a form of use control, tagging may also be extended to allow for network information flow tracking for auditability purposes (Demchenko et al., 2014; Bernard, 2007; Hauser, 2013).

A further method for the protection of controlled data during storage is through the use of encryption. However, the encryption of data requires further consideration. For example, the metadata of encrypted information should still remain exposed in order to allow for functions such as data discovery and retrieval. Other methods of securing data storage involve the fragmentation and dispersal of coherent pieces of data across several nodes. If a single device is compromised in this situation, then only non-coherent fragments of data are equally compromised. Finally, for idempotence during storage, the P2P networking technologies of Subsection 6.2 may assist in the replication of data pieces across nodes. Said technologies are also useful in organising, searching, retrieving, and reassembling fragmented data (Demchenko et al., 2014).

The final phase of data control to be discussed is related to the destruction of data. Due to data retention policies associated with infrastructure-specific regulatory commissions, legal frameworks, or the needs of the enterprise itself, extensive amounts of sensitive information related to the system topology, execution orders, and integrity audit logs, for example, may need to be collected and stored meticulously. Once the properties of the data exceed the policies in place, the nature of such information requires that a clear data destruction policy be in place in order to ensure its irreparable deletion. In certain industries, premature, incomplete or insufficient data destruction may lead to hefty fines for the enterprise upon discovery (Smallwood, 2014).

Strategies for destruction include software techniques, such as cryptographic erase (CE), as well as physical ones, such as degaussing magnetic devices. However, as per the US National Institute of Standards and Technology (NIST), the official position of the US Government is that a single pass over the data using a fixed pattern is sufficient to hinder the retrieval of data, even in the face of state-of-the art forensics technology (Kissel et al., 2014). This stance has been validated by Wright et al.'s research in (Wright and Kleiman, 2008). However, as was the case with the previous phases, context here also plays an important role. To exemplify, bad sectors are normally inaccessible by software. In order to destroy data located in these sectors, different methods are necessary. For a device compliant

with the ANSI ATA and SCSI disk drive interface specifications, one could issue an ATA secure erase (SE) firmware command. This however comes at the cost of having to trust the firmware since the specification does not necessitate reporting or verification. For non-compliant devices, as would be the case for PLCs or embedded devices with on-board memory, other methods would be necessary. Sanitisation techniques engrained in the system and formalised data destruction policies should therefore be carefully designed to match the hardware typically found in the system (Reardon et al., 2014).

8 Conclusions

This paper has focused on discussing the main requirements involved in the design of middleware solutions for vertical integration in industrial enterprise systems. These factors were considered in light of contemporary technological advancements, and solutions selected and examined in order to introduce such aspects as plug-and-play autonomy, portability, interoperability, and dynamic reconfiguration to middleware systems.

The concepts presented show that the design of an ERA-conscious system capable of autonomous integration with existing architectures, or of forming the central core of new architectures, is currently possible with the technologies at hand. Furthermore, as a result of completed and ongoing efforts in protocol standardisation and pursued interests in converging IT and OT technologies, middleware systems may now expand beyond their typical duty of integrating and communicating with legacy devices. Such systems currently have the opportunity to provide the entire enterprise with compartmentalised, idempotent, recoverable, secure, and SOA governance-ready infrastructure. Effectively, the opportunity exists that would allow the ushering forth of a next generation of GSBs; a generation of systems capable of providing the mixed critical processes found in industrial firms with seamlessly omnipresent, business-enhancing, and dynamic services.

Acknowledgements

This paper is supported by TU Wien research funds.

References

Adolphs, P., Bedenbender, H., Dirzus, D., Ehlich, M., Epple, U., Hankel, M., Heidel, R., Hoffmeister, M., Huhle, H., Karcher, B., Koziolek, H., Pichler, R., Pollmeier, S., Schewe, F., Walter, A., Waser, B. and Wollschlaeger, M. (2015) *Reference Architecture Model Industrie 4.0 (RAMI4.0)*, VDI/VDE Society Measurement and Automatic Control (GMA), Düsseldorf, Germany.

Ando, R., Zhang, Z-H., Kadobayashi, Y. and Shinoda, Y. (2009) 'A dynamic protection system of web server in virtual cluster using live migration', *Proceedings of the 8th IEEE International Conference on Dependable, Autonomic and Secure Computing*, pp.95–102.

AutomationML (2014) *Whitepaper AutomationML Part 1 – Architecture and General Requirements*, AutomationML consortium.

Avanzini, A., Valente, P., Faggioli, D. and Gai, P. (2015) 'Integrating Linux and the real-time ERIKA OS through the Xen hypervisor', *Industrial Embedded Systems (SIES), 10th IEEE International Symposium on*, IEEE, pp.1–7.

B&R (2015) *B&R Supports OPC Foundation's Real-Time Working Groups* [online] http://www.br-automation.com/en/company/press-room/br-supports-opcfoundations-real-time-working-groups/ (accessed 18 June 2015).

Basile, F., Chiacchio, P. and Gerbasio, D. (2013) 'On the implementation of industrial automation systems based on PLC', *IEEE Transactions on Automation Science and Engineering*, Vol. 10, No. 4, pp.990–1003.

Bernard, R. (2007) 'Information lifecycle security risk assessment: a tool for closing security gaps', *Computers & Security*, Vol. 26, No. 1, pp.26–30.

Bregenzer, J. and Adämmer, F. (2010) 'Evaluation of integration approaches in common COTS hypervisors for use in industrial automation controllers', *Workshop on Isolation and Integration for Dependable Systems (IIDS)*, ACM.

Byres, E., Carter, J., Franz, M., Henning, W., Karsch, J. and Pederson, D. (2007) *OPC Security White Paper #1 Understanding OPC and How it is Deployed*, Digital Bond, British Columbia Institute of Technology, Byres Research.

Christodorescu, M., Sailer, R., Schales, D.L., Sgandurra, D. and Zamboni, D. (2009) 'Cloud security is not (just) virtualization security: a short paper', *Proceedings of the ACM Workshop on Cloud Computing Security*, *CCSW*, ACM, Chicago, Illinois, USA, pp.97–102.

Colombo, A.W., Karnouskos, S. and Bangemann, T. (2014) 'IMC-AESOP outcomes: paving the way to collaborative manufacturing systems', *Proceeding of the 12th IEEE International Conference on Industrial Informatics (INDIN)*, pp.255–260.

Dai, W., Vyatkin, V., Christensen, J.H. and Dubinin, V.N. (2015) 'Bridging service-oriented architecture and IEC 61499 for flexibility and interoperability', *IEEE Transactions on Industrial Informatics*, Vol. 11, No. 3, pp.771–781.

Demchenko, Y., Ngo, C., de Laat, C., Membrey, P. and Gordijenko, D. (2014) 'Big security for big data: addressing security challenges for the big data infrastructure', in Jonker, W. and Petkovic, M. (Eds.): *Secure Data Management*, Vol. 8425, pp.76–94, Springer International Publishing, Cham.

Didier, P., Macias, F., Harstad, J., Antholine, R., Johnston, S.A., Piyevsky, S., Schillace, M., Wilcox, G., Zaniewski, D. and Zuponcic, S. (2011) *Converged Plantwide Ethernet (CPwE) Design and Implementation Guide*, Cisco Systems, San Jose, California and Rockwell Automation, Milwaukee, Wisconsin.

Driscoll, D. and Mensch, A. (2009) *Devices Profile for Web Services (DPWS) Version 1.1*, OASIS Standard.

Farwick, M., Breu, R., Hauder, M., Roth, S. and Matthes, F. (2013) 'Enterprise architecture documentation: empirical analysis of information sources for automation', *Proceedings of the 46th IEEE Hawaii International Conference on System Sciences (HICSS)*.

Giachetti, R.E. (2011) *Design of Enterprise Systems: Theory, Architecture, and Methods*, CRC Press, Florida, USA.

Gu, Z. and Zhao, Q. (2012) 'A state-of-the-art survey on real-time issues in embedded systems virtualization', *Journal of Software Engineering and Applications*, Vol. 5, No. 4, pp.277–290.

Hauder, M., Matthes, F. and Roth, S. (2012) 'Challenges for automated enterprise architecture documentation', *Trends in Enterprise Architecture Research and Practice-Driven Research on Enterprise Transformation*, Springer, pp.21–39.

Hauser, C. (2013) *A Basis for Intrusion Detection in Distributed Systems using Kernel-level Data Tainting*, PhD thesis. Queensland University of Technology.

He, D., Lobov, A., Moctezumas, L.E.G. and Lastra, J.L.M. (2012) 'An approach to use PERA in enterprise modeling for industrial systems', *IECON – 38th Annual Conference on IEEE Industrial Electronics Society*, IEEE, pp.4196–4203.

Henßen, R. and Schleipen, M. (2014) 'Interoperability between OPC UA and AutomationML', *Procedia CIRP*, Vol. 25, pp.297–304.

Hock, C., Makuth, J., Klostermeyer, A., Jammes, F., Mensch, A. and Colombo, A. (2009) *Deliverable D3.3: Mapping of DPWS/OPC UA/ DPUA into Wireless Nodes and/or Wireless Network Mapping of DPUA into Wired and/or Wireless Network*, SOCRADES [online] http://www.socrades.net/Private/objects/file1237193294.pdf (accessed 5 May 2015).

Jung, M. (2014) *An Integration Middleware for the Internet of Things*, PhD thesis, Vienna University of Technology.

Kastner, W., Kofler, M., Jung, M., Gridling, G. and Weidinger, J. (2014) 'Building automation systems integration into the internet of things the IoT6 approach, its realization and validation', *Emerging Technology and Factory Automation (ETFA), IEEE*, pp.1–9.

Kissel, R., Regenscheid, A., Scholl, M. and Stine, K. (2014) *NIST Special Publication 800-88 Revision 1: Guidelines for Media Sanitization*, Tech. rep., US Department of Commerce, National Institute of Standards and Technology.

Madhavapeddy, A., Mortier, R., Rotsos, C., Scott, D., Singh, B., Gazagnaire, T., Smith, S., Hand, S. and Crowcroft, J. (2013) 'Unikernels: library operating systems for the cloud', *ACM SIGPLAN Notices*, ACM, Vol. 48, pp.461–472.

Mahnke, W., Leitner, S-H. and Damm, M. (2009) *OPC Unified Architecture*, Springer Berlin Heidelberg.

Ngo, H.G. (2013) *From Inter-connecting P2P Overlays to Co-operating P2P Systems*, PhD thesis, Universite Nice Sophia Antipolis; Hanoi University of Sciences.

NSA (2010) *A Framework for Assessing and Improving the Security Posture of Industrial Control Systems (ICS)*, Tech. rep. National Security Agency.

OMG (2005) *Realtime CORBA Specification*, 1.2. Object Management Group, Massachusetts, USA.

OMG (2014) *The Real-time Publish-Subscribe Protocol (RTPS) DDS Interoperability Wire Protocol Specification Version 2.2*, Object Management Group Standard.

Ou, Z. (2010) *Structured Peer-to-peer Networks: Hierarchical Architecture and Performance Evaluation*, PhD Thesis, University of Oulu, Finland.

Phuoc, H.C., Hunt, R. and McKenzie, A. (2008) 'NAT traversal techniques in peer-to-peer networks', *Proceedings of the New Zealand Computer Science Research Student Conference (NZCSRSC).*

Radvanovsky, R. (2013) *Handbook of SCADA/Control Systems Security*, CRC Press, Hoboken, NJ.

Reardon, J., Basin, D. and Capkun, S. (2014) 'On secure data deletion', *Security Privacy, IEEE*, Vol. 12, No. 3, pp.37–44.

Redondo, R., Vilas, A., Cabrer, M., Arias, J., Duque, J. and Solla, A. (2008) 'Enhancing residential gateways: a semantic OSGi platform', *IEEE Intelligent Systems*, Vol. 23, No. 1, pp.32–40.

RTI (2009) *Comparison of OPC and RTI Data Distribution Service (DDS)*, Real-Time Innovations (RTI) [online] http://www.rti.com/docs/RTI_DDS_ and_OPC.pdf (accessed 16 March 2015).

Schmidt, D.C. and Hag, H.v. (2008) 'Addressing the challenges of mission-critical information management in next-generation net-centric pub/sub systems with OpenSplice DDS', *IEEE International Symposium on Parallel and Distributed Processing, IPDPS*, pp.1–8.

Smallwood, R.F. (2014). *Information Governance: Concepts, Strategies, and Best Practices*, Wiley CIO Series, Wiley, Hoboken, New Jersey.

Sobel, W. (2015) *MTConnect Standard*, The Association for Manufacturing Technology [online] http://www.mtconnect.org/standard-documents (accessed 7 August 2015).

Sucic, S., Bony, B. and Guise, L. (2012) 'Standards compliant event-driven SOA for semantic-enabled smart grid automation: evaluating IEC 61850 and DPWS integration', *IEEE International Conference on Industrial Technology (ICIT)*, pp.403–408.

Thramboulidis, K. (2013) *IEC 61499 vs. 61131: A Comparison Based on Misperceptions*, in arXiv preprint arXiv:1303.4761.

Ungurean, I., Gaitan, N.C. and Gaitan, V.G. (2014) 'Transparent interaction of SCADA systems developed over different technologies', *18th International Conference on System Theory, Control and Computing (ICSTCC)*, pp.476–481.

Wales, C. (2015) *Middleware and Related Services (MARS) Platform Task Force (PTF)* [online] http://www.omgwiki.org/mars/doku.php?id= start (accessed 30 June 2015).

Wilamowski, B.M. and Irwin, J.D. (2011) *Industrial Communication Systems*, The Industrial Electronics Handbook, CRC Press, Floriday, USA.

Williams, T.J. (1993) 'The Purdue enterprise reference architecture', *Proceedings of the JSPE/IFIP TC5/WG5.3 Workshop on the Design of Information Infrastructure Systems for Manufacturing*, *DIISM*, North-Holland Publishing Co., Amsterdam, The Netherlands, pp.43–64.

Wright, C. and Kleiman, D. (2008) 'Overwriting hard drive data: the great wiping controversy', *Information Systems Security*, pp.243–257, Springer, Berlin, Heidelberg.

Yoo, S. and Yoo, C. (2014) 'Real-time scheduling for Xen-ARM virtual machines', *IEEE Transactions on Mobile Computing*, Vol. 13, No. 8, pp.1857–1867.

Zerbst, J., Zimmermann, S., Holstein, D.K. and Poirier, C. (2013) 'Towards an adapted classification methodology for graded security approaches in EPU architectures', *Proceedings of the International Council on Large Electric Systems (CIGRE).*