Network Protocols and Algorithms
ISSN 1943-3581
2010, Vol. 2, No. 3

Architectural Evolution of Monitor and Control Systems - Issues and Challenges

Christian Esposito and Domenico Cotroneo

Department of Computer and Systems Engineering,

Università di Napoli - "Federico II"

Napoli, 80125 - Italy

E-mail: {christian.esposito, cotroneo}@unina.it

Aniruddha Gokhale and Douglas C. Schmidt

Institute for Software Integrated Systems

Department of Electrical Engineering and Computer Science - Vanderbilt University

Nashville, TN 37235 - USA

E-mail: {a.gokhale, d.schmidt}@vanderbilt.edu

Received: October 11, 2010 Accepted: October 25, 2010 DOI: 10.5296/npa.v2i3.483

Abstract

Large-scale Complex Critical Infrastructures (LCCIs), such as power grids and transport infrastructures (*e.g.*, network of airports and seaports), play a key role into several fundamental human activities, and represent the next generation of Monitor and Control Systems. They make extensive usage of Information and Communications Technology (*e.g.*, computing systems, communication networks, and sensing hardware) for providing support for advanced monitoring and control facilities. However, solutions currently adopted in the field exhibit several inefficiencies when applied to LCCI. On the contrary, Distributed Event Based Systems (DEBS) seem promising solutions due to their intrinsic decoupling properties that enforce strong scalability degrees. However, they also present some open issues when dealing with the challenges imposed by LCCIs. This paper provides an introduction on LCCIs, a presentation of their challenges, and an analysis on the adoption of DEBS for LCCIs.

Keywords: Monitor and Control Systems, Multicasting, Publish/subscribe Services

1. Introduction

Monitor and Control Systems (MCSs) [1] are special-purpose engineering artifacts designed to automatically sense the pulse of a given system and defend it against exogenous events by adjusting its execution parameters. They have been traditionally architected by interconnecting the several components within a MCS by means of dedicated networks, and they provide no, or limited, connectivity to the outside world. However, when applied at an extreme high scale and complexity, such as for Air Traffic Management or Power System Control, such architecture has exhibited several inefficiencies, which motivated the shift to a novel, collaborative, architectural perspective. Specifically, such an innovative architecture envisions large-scale MSCs, referred as **Large-scale Complex Critical Infrastructures** (LCCIs) [2], as architected by means of federating several heterogeneous systems via a certain middleware platform. This represents a novel perspective on how next generation MCSs are realized: a shift in scale and in the structure, from monolithic and vertical architectures, which characterized traditional systems, toward large, highly modular and integrated systems. However, the challenges introduced by such a novel perspective go beyond what current systems, such as *Supervisory Control and Data Acquisition* (SCADA), can deliver. Therefore, there is a need to foster the further development of innovative middleware technologies in the field.

Recently, **Distributed Event Based Systems** (DEBSs) [3] have emerged as an attractive solution for the interconnection of several autonomous systems and as a suitable alternative to SCADA systems for LCCIs. Actually, several innovative industrial projects are developing complex distributed MCSs consisting of Internet-scale federations of heterogeneous entities via middleware solutions. These kinds of systems in application domains, such as Air Traffic Management, are being engineered around a data-centric paradigm, by means of middleware platforms based on the publish/subscribe interaction model, which owns intrinsic decoupling properties and implicit multicast capabilities that enforce the scalability properties provided by the middleware. Beside the appealing advantages of using publish/subscribe DEBSs in LCCIs, there are still several open issues that need to be investigated and resolved.

This introduction to the issue aims to *(i)* discuss the architectural evolution that we are witnessing in the field of MCSs; *(ii)* identify the novel challenges caused by the extensive usage of Information and Communications Technology (ICT) made by LCCI; *(iii)* introduce DEBSs, publish/subscribe interaction model and their features; and *(iv)* survey open issues that DEBSs exhibit when facing the challenges imposed by LCCI. We then conclude by presenting the articles in the special issue.

2. Monitor and Control Systems

MCSs are typically composed of three main components, as depicted in Figure 1: i) *Sensors*, which monitor the behaviour of the system or process under control, ii) a *Controller*, which returns the commands obtained by the execution of some specific control algorithms based on the data received by the sensors, and iii) *Actuators*, which transform the commands received by the controller in proper control actions performed on the system or process under control. Optionally, there may be a fourth component, namely a Human Interface, which collects statistics from the controller to provide a human operator with a view of the state and behaviour of the system under control.

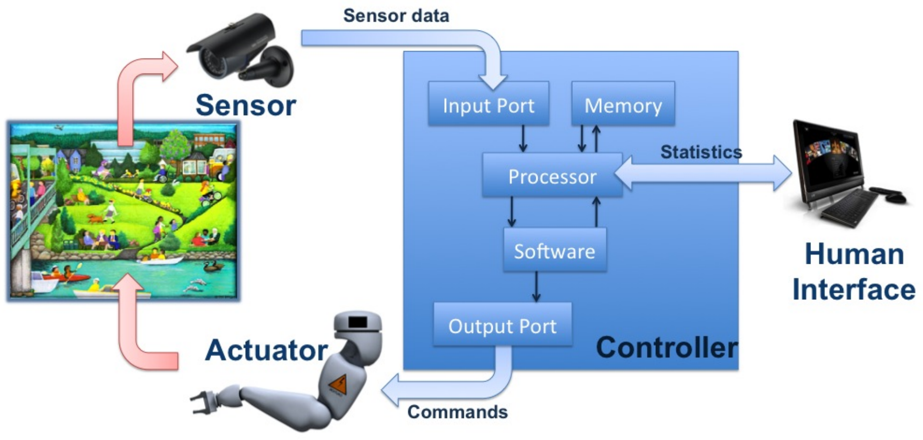

Figure 1 - Basic architecture of an MCS

Since there is a huge amount of information exchanged among the previous components, the success of MCSs depends on how effectively data are distributed. For example, if monitoring data do not reach the controller on time, or even worse they are lost, the MCS is not able to perform the related actions properly, leading to damages or even considerable loss of human lives and/or money. Through the years we have witnessed an evolution of how MCSs are architected, *i.e.*, how sensors, controller and actuators are interconnected, and the scope of this section is to provide a brief description of such an evolution. Specifically, Section 1.1 provides a description of the traditional architecture adopted by current MCSs, whereas Section 1.2 introduces the novel architecture that has been used to develop the next generation of MCSs.

1.1 Traditional "closed" architecture

The typical communication topologies for early MCSs, which have been successfully adopted for decades, consist of point-to-point connections among sensors, controller and actuators [4]. Despite still in use for very simple MCSs, such as automated teller machines or pacemakers, such a centralized point-to-point approach has been proved to be unsuitable for complex MCSs, such as the governance of battleships or aircraft flight control. In fact, it exhibits severe scalability, reliability and maintainability limits and is not able to satisfy requirements of complex MCSs, such as modularity, decentralized control, quick and easy maintenance and low cost.

To overcome limitations of point-to-point connections, a new approach [5], which gave birth to a novel generation of MCSs called **Networked Control Systems** (NCSs) [6], has been introduced. Specifically, in an NCS, sensors, controller and actuators are no more glued together through dedicated point-to-point connections but through common-bus network architectures, such as Ethernet or a Local Area Network (LAN). The adoption of this networked approach has improved efficiency, flexibility and reliability and reduced installation, reconfiguration and maintenance costs [4]. Currently, NCSs are often implemented as SCADA systems (as depicted in **Figure 2**) [7], so that over time the two terms have become synonymous. SCADA systems are composed of the following components:

- A *Human-Machine Interface* (HMI), which allows a human operator to monitor and control a process;
- A *Supervisor Station*, which gathers data on the process under control and sends commands;
- Several *Remote Terminal Units* (RTUs), which connect to the physical equipment by converting electrical signals from devices to digital values for the supervisor station or vice versa.

It is simple to note a direct mapping between the components of a generic MCS and the ones of a SCADA system. In fact, the Supervisor Station is equivalent to a controller, while RTUs plays the role of both sensors and actuators. In the first case, RTUs feed the Supervisor Station with data about the current state of the process under control, while in the second case they convert and send out commands to the equipment.

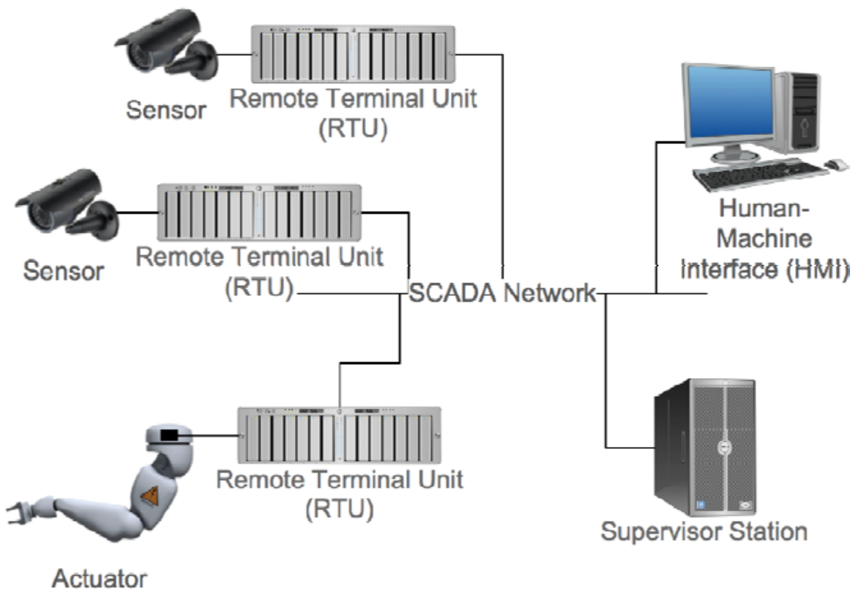
Sensor
Remote Terminal Unit (RTU)
Sensor
Remote Terminal Unit (RTU)
SCADA Network
Human-Machine Interface (HMI)
Remote Terminal Unit (RTU)
Actuator
Supervisor Station

Figure 2 - Architectural overview of a SCADA system

The increase in the complexity of NCSs has made it unsuitable to concentrate only on a central component the task of gathering all sensor data and processing them to infer commands to maintain a process within a desired behaviour [8]. In recent SCADA systems a centralized supervisor station is replaced by multiple interconnected stations, each one responsible for a specific control task. Therefore, nowadays SCADA systems are also used to indicate the so-called **Distributed Control Systems** (DCSs) [9], *i.e.*, systems that distribute control in multiple devices rather than a single centralized one that control all the system. A practical example is provided by the "Fly-by-Wire" systems [10] that assist pilots to control the flight of an aircraft. In fact, there is no single control station in such a NCS, but several stations are placed in different parts of the aircraft and are in charge of monitoring and controlling a specific aspect of the flight, such as altitude, speed, collision detection, fuel capacity and so on. The command console of the aircraft includes gauges and controls so that pilots can coordinate the actions that each control system may take.

Due to security and availability concerns, the NCS architecture is traditionally based on a monolithic, "*closed world*" perspective, *i.e.*, several computing nodes interconnected by dedicated networks with limited or no connectivity to the outside world. Therefore, frameworks that have to carry out complex control activities, such as the *Air Traffic Management* (ATM) or the *Power System Control* (PSC), have been fragmented into "islands of automation", *i.e.*, they are composed by several systems, each one in charge of controlling an isolated portion but with no interaction with any other system. The current ATM framework in Europe [11] provides a concrete example of such architecture. In fact, it is structured into several systems, called *Area Control Centers* (ACC), each one in charge of performing en-route and landing/departing control of aircrafts in their assigned sector of the airspace. However, a given ACC does not directly exchange information with any other neighboring ACCs. In fact, each ACC can monitor flights in its assigned portion of the airspace, without having any information of flights in other portions. When the control of a flight has to be handled from an ACC to another one, the only way to exchange information is by means of radio communications among human operators. Similarly, the management of national PSCs is organized in a similar "*closed*" architecture [12]. Indeed, a national PSC consists of ACCs, which are in charge of managing a restricted area of the PSC and are coordinated by *Regional Control Centers* (RCCs) and a single *National Control Centre* (NCC). However, there is no direct connection between NCCs of different countries, but, if necessary, human operators will interact each other only via radio communications.

1.2 Novel "open" architecture

The traditional "closed world" perspective has been proven to be affected by severe limitations and inefficiencies, and the following ones are just some concrete examples:

- As stated by the EUROCONTROL Performance Review Commission (PRC) in one of its last reports (PRR 2006 - [13]), the fragmentation of European airspace affects flight efficiency and limits the ability of the en-route function to support airport throughput with an estimated additional cost of €1.4 billion per year. These inefficiencies have also a direct negative impact on the environment, given the increased level of emissions produced (according to PRR 2006, 4.7 million tons of CO2 per annum);
- National NCCs can be no more viewed as closed systems, since a fault in a national power grid has high probability to negatively affect the power grid of another country due to their massive interconnection and interdependency[1] (*e.g.*, a severe blackout happened in Italy in October 2001, which lasted almost for an

[1] For example, less than 10% of the Italian electricity demands during the daylight and picks of 25% during the night are covered by electricity imported from neighboring countries such as France and Switzerland.

entire day, due to a power line in Switzerland that had gone haywire).

The demand for more complex and efficient MCSs and the extraordinary success of the Internet as communication channel are causing an evolution of NCSs, leading to a new, third, generation of MCSs, called **Large scale Complex Critical Infrastructures** (LCCIs), which represent an example of the **Ultra Large Scale** (ULS) systems that has been envisioned in a report produced by Carnegie Mellon University's Software Engineering Institute (SEI) in June 2006 [14]. As shown in **Figure 3**, such systems adopt an innovative "*open world*" architecture that consists of a dynamic Internet-scale hierarchy/constellation of interacting heterogeneous systems, which cooperate to perform critical functionalities. LCCIs represent a solution to the unsuitability of traditional architectures, and to the urgent need for more integrated control architectures. In fact, the federation that characterizes LCCIs goes beyond the trivial monitoring data exchanges between distinct control systems, but allows implementing complex decentralized and distributed control algorithms where decisions are taken by the cooperation of several controllers geographically distributed. Such collaborative intelligence underlying the control decision-making process is necessary since control decisions taken in a given portion of a critical infrastructure may affect the other portions, as noticed in the power grid collapse event previously described.

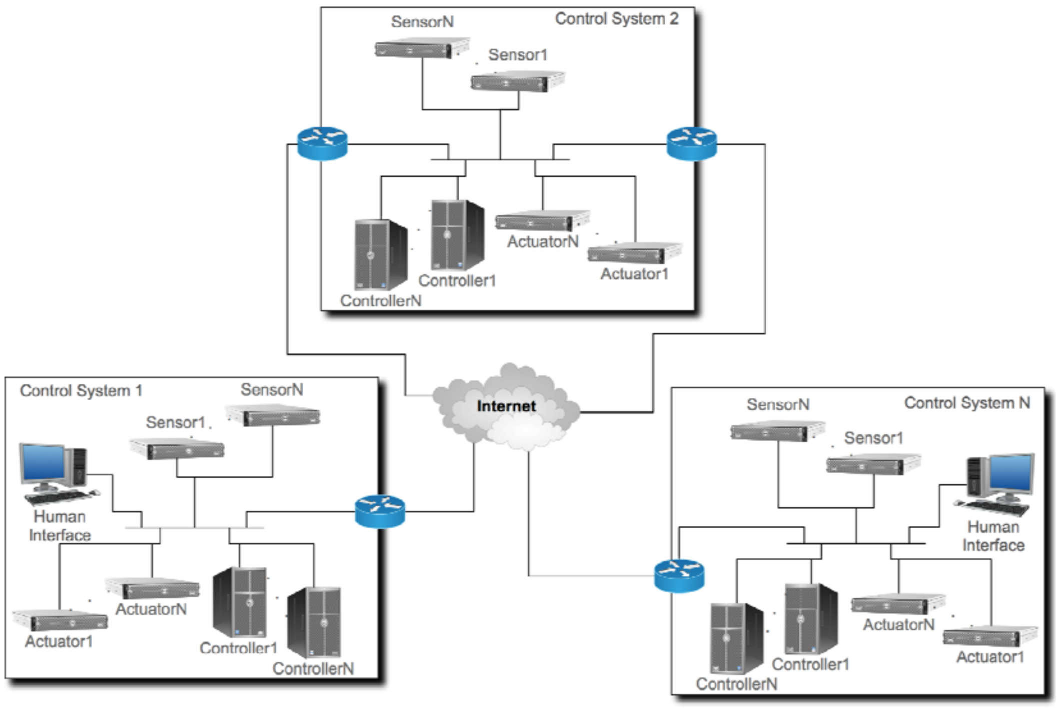

Figure 3 - Architectural overview of a generic LCCI

Many of the ideas behind LCCIs are increasingly "in the air" in several current projects that aim to develop innovative MCSs. For example, EuroCONTROL has funded a project to devise the novel European ATM framework in Europe, called *Single European Sky ATM Research* (SESAR)[2], which aims to develop a seamless infrastructure to allow control systems to cooperate in order to have a wider vision of the airspace and better handle the

[2] www.eurocontrol.int/sesar/public/subsite_homepage/homepage.html

growing avionic traffic in Europe. Specifically, the different ACCs of the current ATM framework are going to be federated via a middleware solution, so to exchange ATM information and better orchestrate ATM activities. On the other hand, more and more research in the field of power grids focuses on new control paradigms [15],[16] that step away from the traditional centralized control paradigms by introducing the concept of *Microcells*, which are small-scale versions of today's huge centralized electricity system. Like a centralized power grid, microcells can generate, distribute and regulate the flow of electricity to consumers, and their dimensions, smaller than traditional power grids, imply the advantage of a more precise power control. In addition, microcells are also networked among each other so as to achieve a smoother power regulation within the overall microcell federation, by making regulation activity based on monitoring data within a given microcell and also on data received from other microcells.

This novel perspective is also enforcing the integration of several different kinds of IT infrastructures that are usually strictly distinct and separated from MCSs. A practical example is provided by the EU project called *Total Airport*[3], whose scope is the integration of all sub-systems for managing land-side and air-side activities and their information flows. Specifically, critical systems and the relative federation middleware under development within the context of SESAR are going to be integrated with all the IT components used for airport management, and also with ubiquitous systems for safe and secure passenger and luggage management, so as to realize a seamless "*door-to-door*" control (*i.e.*, from entering the departing airport until leaving the arriving airport). For example, passengers can have access to certain ATM information via their smart phones to know the status or schedule of their flights, to track their luggage, to locate themselves within the airport map or even to receive commercial ads.

The adoption of a federated architecture *1)* raises the scale of traditional MCSs, *2)* makes possible complex interaction patterns among the elements of LCCIs and *3)* requires the use of unmanaged networks, such Wide-Area Networks (WAN) and/or wireless ad-hoc networks, rather than dedicated LAN. This brings several novel challenges on the data dissemination infrastructure used to implement the federation:

- *Scalable Event Dissemination* - The scale of LCCIs represents a serious challenge when designing such systems and must be taken into high consideration. The time to deliver a message does not have to be strongly affected by such a scale, both in terms of devices composing the system (vertical scale) and the traffic generated by the data sources (horizontal scale).
- *Reliable and Timely Event Delivery*
    - o MCSs have been used in several different domains, spanning from military applications (*e.g.*, defending ships against missile attacks or controlling unmanned combat air vehicles through wireless links) to civil applications (*e.g.*, regulating the temperature of coolant in nuclear reactors and

[3] www.eurocontrol.int/eec/public/standard_page/EEC_News_2006_3_TAM.html

maintaining the safe operation of steel manufacturing machinery). All these heterogeneous use cases have one feature in common: the right answer delivered too late becomes the wrong answer [17]. Since the reactivity of embedded systems depends on the time needed to exchange information among its components, it is crucial that the adopted communication infrastructure copes with timing failures by guaranteeing a timely delivery of data produced by the several elements in an LCCI (*Timeliness*).

- o The adopted networking infrastructure can exhibit a faulty behavior, *i.e.*, several failures can impose message losses, as shown for Internet by [18] and [19] or for Wireless Networks by [20], or applications can face failure manifestations, such as crashes or hangs. Due to the critical nature of LCCIs, it is crucial that exchanged data reach their destinations despite of failure manifestations by using one of the well-defined fault-tolerance strategies available in literature (*Reliability*).

LCCIs require the adoption of a proper delivery strategy so to achieve both reliability and timeliness in the event notification, *i.e.*, delivering messages within given temporal deadlines even if several failures may occur.

- *Adaptive Event Dissemination* - As mentioned, LCCIs are composed of heterogeneous devices interconnected by different kinds of networks, each affected by different behavioural patterns. In fact, this variety of networks implies that network conditions are not the same all over the infrastructure. Moreover, wireless networks are subject to a high variability of the network conditions due to the variability of the interference that can affect the workspace. This means that *1)* one solution does not fit all, and *2)* the dissemination middleware cannot be tuned at configuration time. On one hand, only one reliability strategy to guarantee event dissemination cannot be embodied into the middleware, but it has to be able to use the proper strategy depending on the measured condition of the network, in terms of experienced loss probability and burst length. Therefore, within the system, several reliability strategies may be needed in the different portions of the infrastructure that do not share the same network conditions. On the other hand, due to the high variability of the network conditions, the middleware has to expose autonomic capability to self-tune and self-heal itself when the network conditions change. This issue belongs to the wider requirement of *Adaptability*, *i.e.*, a publish/subscribe middleware has to optimize their computation based on current contextual situations (*e.g.*, resources available on the user device, network bandwidth). Adaptation means essentially to properly detect the frequent changes that may happen in the system, and react to changes by reconfiguring the system accordingly.

- *Flexible Event Notification* - LCCIs are rarely built ex-novo, but it is more probable that they are developed starting from already-existent legacy systems by using a middleware and other proper abstractions to federate them. Federating legacy systems built by different companies at different times and under different regulation laws raises a problem known in the database community as the *Data Exchange Problem* [21]. Specifically, the data dissemination infrastructure has to provide flexibility so that communication participants with different views of the exchanged data can understand each other.

- *Secure Event Dissemination* - Since LCCI communications are performed over unmanaged networks, it is crucial that the data distribution infrastructure protects exchanged information from misuse by malicious users so to provide the following security assurances: *1)* only legitimate destinations are able to quickly and efficiently access information of interest, *2)* improper changes to exchange data are inhibited, and *3)* unauthorized sources are prevented from exchanging information with other participants of the LCCI. Therefore, security is not only intended as preventing unauthorized reading of data, but also avoiding improper modifications or updates.

3. Distributed Event-Based Systems

LCCIs exhibit features and requirements that go beyond what current SCADA/DCS systems can provide. Therefore, they require a novel data dissemination approach, and the so-called **Distributed Event-Based Systems** (DEBSs) [3] represent promising candidates due to their intrinsic asynchrony and decoupling characteristics. In fact, they are already been using in large-scale enterprise applications and for federating critical systems. For example, in the context of SESAR, EuroCONTROL has selected them as a key technology for the interconnection of ACCs.

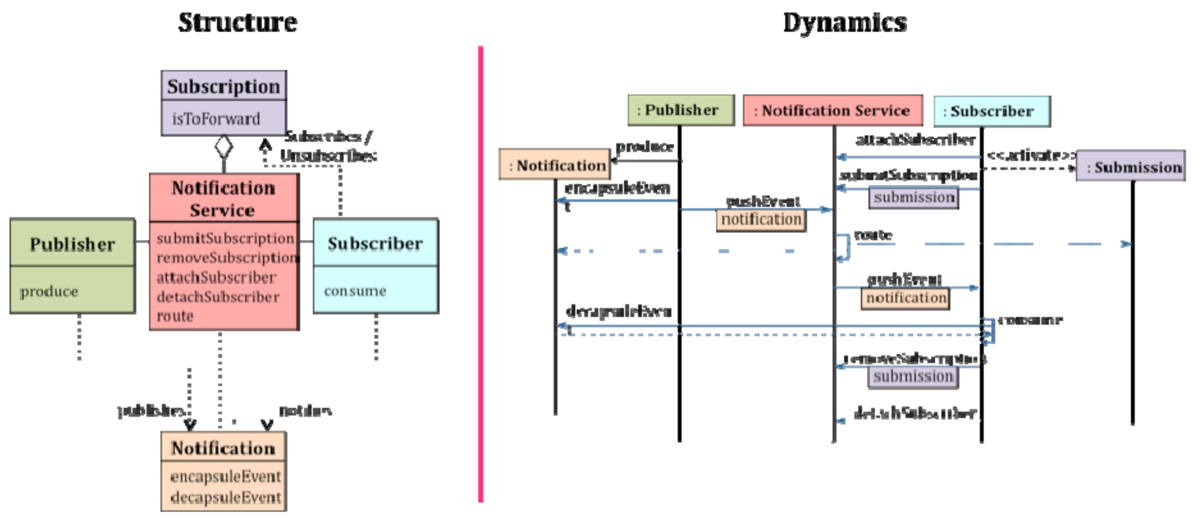

Figure 4 - Structure of a generic publish/subscribe DEBS and operations needed to deliver a notification from a publisher to a subscriber.

DEBSs denote information as *events*, which are detectable conditions in the application, *e.g.*, any changes in its state, while the act of delivering an event is indicated as *notification*. Such middleware solutions are based on the popular design pattern called **Publish/Subscribe**

[22], which defines an interaction model composed of three key entities, depicted in **Figure 4**:

1. *Publishers* - applications that generate notifications;
2. *Subscribers* - application that consume notifications in which they have expressed interest by means of so-called *Subscription*;
3. *Notification Service* - entity that glues together publishers and subscribers by allowing notifications to flow from the first ones to the second ones.

As shown in the right side of **Figure 4**, an application can receive notifications only after it has contacted the Notification Service to give its reference and has defined a proper Subscription to submit at the Notification Service. Whenever an event occurs at a Publisher, it generates a Notification and encapsulates the event in it. Then, the Notification is sent to the Notification Service, which routes it to the proper Subscribers depending on the Subscriptions it has stored. A Subscriber consumes a received Notification by retrieving the contained event and processing it. Before leaving the system, a Subscriber has to remove its Subscription and detach itself from the Notification Service. The mediator role of the Notification Service allows DEBSs to support decoupling properties in space, time and synchronization [23], so that the communication is anonymous, non-blocking, and asynchronous. Decoupling production and consumption of notifications has been proven to enforce scalability [24] since all explicit dependencies between the participants to event-based communications are removed. These offered decoupling properties make the resulting communication infrastructure well adapted to large-scale distributed environments.

A significant amount of publish/subscribe systems have been developed and implemented, both by industry and by academia [25]. It is possible to group these different implementations according to the adopted *Subscription Model*, i.e., how Subscriptions are expressed, and *Notification Architecture, i.e.*, how the Notification Service is implemented. In the first case, considering the subscription model, we have the following classification:

1. *Topic-based* DEBS: participants can publish and subscribe to individual subjects;
2. *Content-based* DEBS: notifications are subscribed according to a matching function on their content;
3. *Type-based* DEBS: notifications belong to a specific type, encapsulating attributes as well as methods.

On the other hand, we can have the following taxonomy of notification architectures:

1. *Direct Architectures*: each Publisher node takes care of routing published notifications;
2. *Centralized Architectures*: a broker placed between publishers and subscribers implements the functionalities of the Notification Service;
3. *Distributed Architectures*: The Notification Service is devised as a network of

brokers.

The protocols for implementing the notification architectures are traditionally grouped in two main classes [26]: *Transport-Level Multicast* (TLM) and *Application-Level Multicast* (ALM). As the name suggests, TLM protocols devise the mechanism for multi-point content delivery at the Transport Layer of the ISO/OSI stack by adopting IP Multicast. On the other hand, ALMs do not use IP Multicast but implement the multicasting service at the application layer: end-systems are interconnected using an overlay network and the duty of replicating a message when it has to be dispatched over several distinct outgoing links is carried out by end-systems rather than routers. Concrete examples of the use of ALM protocols to implement the notification service are the topic-based *Scribe* [27] and content-based *Siena* [28].

The key architectural question is which paradigm is more suitable to implement an effective multicasting: ``to be or not to be" at the transport layer? Unfortunately, there is no simple answer since the choice strongly depends on the requirements that the multicast protocol has to address. In fact, an architecture perspective may fit some use cases but does not other ones. ALM is less efficient than IP Multicast since it exhibits higher link stress. Therefore, IP Multicast provides a wiser use of the network resources, by reducing the traffic, and better performances that any ALM protocol. Although IP Multicast outperforms ALM, it is a mistake to think that it represents the best solution for every application scenario. In fact, the adoption of IP Multicast has been dogged by several drawbacks. ALMs present neither the deployment issues neither the scaling limitations that affect IP Multicast over wide area networks [29].

4. Open Issues

As mentioned, DEBSs are appealing solutions for architecting LCCIs since they natively offer strong scalability properties. In addition, the publish/subscribe interaction model provide the abstraction of a *Global Data Space*, which allows elements to easily exchange information in a data-centric manner by hiding underlying routing concerns, and to adopt a *plug-and-play* perspective, which allows new components to be easily included in the running LCCI without turning it down ("*hot*" inclusion of new elements). These features are crucial since they strongly simplify the management of large-scale federations such as LCCIs. However, there are still several open issues to completely satisfy the requirements presented in Subsection 1.2:

- *Challenges for Reliable and Timely Event Delivery* - In event-based architectures, reliability can refer to two aspects. On one hand, the availability of the event dissemination service is resilient to the volatile nature of the end-hosts due to the occurrence of failures or churns. On the other hand, the reliable delivery of event content from publishers to subscribers despite possible losses caused by the network and/or the limited size of the sending/receiving buffers on the end-hosts. The first aspect has been widely investigated in the past and several approaches are available [30],[31]. However, less attention has been paid to the issue of guaranteeing event dissemination in a reasonable time [32], so current publish/subscribe platforms fail to provide both reliable and timely event dissemination. In fact, the reliability gain is always obtained at the expenses of predictable and stable performance, such as in case of retransmission-based recovery schemas such as *Gossiping* described in [33]. On the other hand, the only approach that enforces timeliness, *i.e.*, spatial redundancy-based recovery schemes, such as *Forward Error Correction* (FEC) described in [34], exhibits severe scalability limitations due to the centralization of the recovery operations at the publisher.

- *Challenges for Adaptive Event Dissemination* - There is a rich literature on self-adaptive systems. In most works on adaptive systems, planning is task-based where the designer of adaptation policies is required to describe all the possible system configurations [35], and planning becomes equivalent to solving an optimization problem of finding the best configuration. Alternative approaches have started to emerge, such as using AI techniques [36]. However, they are still in early stages and their application to publish/subscribe middleware still needs to be established.

- *Challenges for Flexible Event Notification* - There are several solutions to introduce flexibility in current publish/subscribe middleware. However, all of them cannot be applied in the context of LCCIs, since they offer flexible notifications at the expenses of delivering data at the right time. The widely used solution to offer flexibility is to adopt XML as serialization format. However, XML redundant syntax strongly affects the delivery latency, as shown in [37]. On the other hand, the publish/subscribe middleware that are largely adopted in critical scenarios, such as the ones compliant with the OMG standard called Data Distribution Service (DDS) [38], pay more attention to timeliness by using serialization formats that minimize the bytes exchanged through the network, with the advantage of reducing delivery time but with the drawback of compromising flexibility by constraining applications to adhere to predefined data structures.

- *Challenges for Secure Event Dissemination* - The issue of achieving secure event dissemination cannot be trivially addressed by using solutions applicable to point-to-point communication. On one hand, when applied to multicast communication, such solutions exhibit severe scalability and performance limitations. On the other hand, their applicability is discouraged since they form obstacles to key properties of the publish/subscribe model. Classical cryptographic key management requires that the two end-point of a communication agree on shared keys, which are used to encrypt and decrypt the exchanged messages. This violates the decoupling and anonymity properties between publishers and subscribers. Moreover, content-based publish/subscribe middleware intrinsically makes it impossible to adopt the standard cryptographic techniques. In fact, the brokers composing the notification service need to access the event content in order to perform the complex publication-subscription matching, needed to properly forward the received events to the right subscribers, and support anonymity of publishers and subscribers. Although significant amount of work has been done in the field of secure group communication to make key management more scalable [39], it is not possible to apply the results directly. A different perspective is needed in publish/subscribe services because group membership is not as flexible as the submission model of publish/subscribe platforms. Rather then associating keys to users or groups of users, an approach is to associate authorization keys to subscriptions and encryption keys to publications [40]. A concrete implementation of it on top of Siena proved that it allows secure event dissemination by maintaining low latency overhead. However, the main drawback is an explosion of keys that the infrastructure has to manage when increasing its scale. Another approach adopts the emerging paradigm of access control architectures [41]. Specifically, these architectures provide mechanisms for authorizing event clients to publish and subscribe to event types. Their local broker checks the privileges of the end-hosts in order to be able to access the publish/subscribe system. Unfortunately, the approach implements access control at the edge of the broker network and assumes that all brokers can be trusted to enforce the access control policies correctly. So, any malicious, compromised or unauthorized broker is free to read and write any events that pass through it on their way from the publishers to the subscribers. This might be acceptable in a relatively small system deployed inside a single organization, but it is not appropriate in large-scale infrastructures such as LCCIs.

5. DD4LCCI Special Issue

This special issue is based on the Data Dissemination for Large-scale Complex Critical Infrastructure (DD4LCCI) Workshop held at the Eighth European Dependable Computing Conference (EDCC) in Valencia in 2010. The workshop was well attended and characterized by very interactive and constructive discussions. The scope of the workshop was to provide a forum for researchers and engineers in academia and industry to foster an exchange of research results, experiences, and products in the area of reliable, timely and scalable data dissemination in large-scale critical systems from a perspective of middleware support.

The six articles in this special issue will give readers a glimpse of the problem and solution domains in publish/subscribe middleware for LCCIs. They can be broadly classified into two distinct classes: research efforts tailored on addressing the mentioned open issues faced when DEBSs are applied to LCCIs, and practical experiences of using DEBSs in concrete LCCI application domains. Specifically, in the first category we have the following four papers:

1. "Exploring the Performance Tradeoffs among Stability-Oriented Routing Protocols for Mobile Ad hoc Networks" discusses how to efficiently disseminate information on MANET. Through simulations conducted on ns2, it analyzes several

stability-oriented on-demand MANET routing protocols. Therefore, it is related to the challenge of Reliable and Timely Event Delivery;

2. "Integrating Machine Learning Techniques to Adapt Protocols for QoS-enabled Distributed Real-time and Embedded Publish/Subscribe Middleware" investigates the adoption of machine learning technique for adapting the adopted dissemination strategy to the observed network conditions. Therefore, it is related both to the challenge of Reliable and Timely Event Delivery and the challenge of Adaptive Event Dissemination;

3. "Achieving Security by Intrusion-Tolerance Based on Event Correlation" studies the issue of intrusion detection by proposing a multi-level architectural framework, which collects heterogeneous information and event correlation to diagnose intrusion symptoms. Therefore, it is related to the challenge of Secure Event Dissemination;

4. "Dynamic Ontology-Based Redefinition of Events Intended to Support the Communication of Complex Information in Ubiquitous Computing" presents a model of dynamically redefinable events in order to support dynamic reconfiguration, *i.e.*, to adapt events to changes in their structure and participant entities without any need of user intervention. Therefore, it is related to the challenge of Flexible Event Notification.

While, the remaining papers belongs to the second category:

1. "Data distribution technologies in wide area systems: lessons learned from SWIM-SUIT project" shows the application of publish/subscribe middleware technologies to the domain of Air Traffic Management, specifically the novel European framework called SESAR. Specifically, the authors introduce a novel middleware platform, called SWIM-SUIT, whose scope is the efficient interconnection of ACCs within the SESAR framework. They present the unresolved problems within this area and how to address them by means of publish/subscribe middleware;

2. "A Cyber Physical Systems Perspective on the Real-time and Reliable Dissemination of Information in Intelligent Transportation Systems" illustrates another concrete example of LCCI, namely Intelligent Transportation Systems (ITS). In such an application domain the main issue is the timely and reliable dissemination of traffic-related information to car drivers. The authors present the results on their studies on issues related to car-to-infrastructure communication by means of wireless networks.

We hope you will enjoy reading these articles. In closing, we want to thank all those who contributed to the success of the DD4LCCI Workshop in Valencia and who enabled this special issue to be published.

References

[1] P. Marwedel "Embedded System Design", Springer 2006;

[2] S. Bologna, C. Balducelli, G. Dipoppa, and G. Vicoli, “Dependability and Survivability of Large Complex Critical Infrastructures”, Computer Safety, Reliability, and Security, Lecture Notes in Computer Science, volume 2788, pg. 342–353, September 2003;

[3] G. Muhl, L. Fiege and P. Pietzuch, “Distributed Event-Based Systems”, Springer, 2006;

[4] T. C. Yang, “Networked Control System: a Brief Survey”, IEEE Proceedings on Control Theory and Applications, 153(4): 403-412, July 2006;

[5] E. A. Lee, “What’s Ahead for Embedded Software?”, IEEE Computer, pg: 18-26, September 2000;

[6] G.J. Pottie and W.J. Kaiser, “Principles of Embedded Networked Systems Design”, Cambridge University Press, 2008;

[7] E.-K. Chan and H. Ebenhoh, “The Implementation and Evolution of a SCADA System for a Large Distribution Network”, IEEE Transactions on Power Systems, volume 7, number 1, pg. 320-326, February 1992;

[8] F. F. Wu, K. Moslehi and A. Bose, “Power System Control Centers: Past, Present and Future”, Proceedings of the IEEE, volume 93, number 11, pg. 1890-1908, November 2005;

[9] F.-L. Lian, J. Moyne and D. Tilbury, “Network Design Considerations for Distributed Control Systems”, IEEE Transactions on Control Systems Technology, volume 10, number 2, pg. 297-307, March 2002;

[10] D. Briére, D. Ribot, D. Piluad and J.-L. Camus, “Methods and Specification Tools for Airbus On-board Systems”, Microprocessors and Microsystems, volume 19, number 9, pg. 511-515, November 2005;

[11] EuroControl, “Milestone Deliverable D1 - Air Transport framework: The current Situation”, SESAR Library, 2006;

[12] Deliverable D2 of the EU project “Critical Utility InfrastructurAL Resilience” (CRUCIAL), 2006;

[13] Performance Review Commission, “Performance Review Report (PRR) 2006 - An Assessment of Air Traffic Management in Europe during the Calendar Year 2006”, May 2006, available at www.eurocontrol.int/prc/gallery/content/public/Docs/PRR_2006.pdf, accessed in August 2010;

[14] L. Northrop et al., “Ultra-large-scale Systems, The Software Challenge of the Future”, Software Engineering Institute, Carnegie Mellon University, June 2006, available at www.sei.cmu.edu/uls/ accessed in December 2009;

[15] M. Amin and B.F. Wollenberg. “Toward a Smart Grid”, IEEE Power & Energy Magazine, Vol. 3, No. 5, Pg. 34-41, September/October 2005;

[16] P. Lund, S. Cherian and T. Ackermann. “A Cell Controller for Autonomous Operation of a 60kV Distribution Area”, International Journal of Distributed Energy Resources, Vol. 2 No. 2, April-June 2006;

[17] D.C. Schmidt, “Middleware for real-time and embedded systems”, Communications of the ACM, volume 45, issue 6, pg: 43-48, June 2002;

[18] A. Markopoulou, F. Tobagi, and M. Karam, “Loss and Delay Measurements of Internet Backbones”, Computer Communications, 29(10): 1590-1604, September 2003;

[19] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y. Ganjali, and C. Diot, “Characterization of Failures in an Operational IP Backbone Network”, IEEE/ACM Transactions on Networking (TON), 16(4):749-762, August 2008;

[20] M. G. Arranz, R. Aguero, L. Munoz, and P. Mahonen, “Behavior of UDP-based Applications over IEEE 802.11 Wireless Networks”, Proceedings of the 12th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, pg. F-72 - F-77 vol.2, September/October 2001;

[21] P. G. Kolaitis, “Schema Mappings, Data Exchange, and Metadata Management”, Proceedings of the ACM Symposium on Principles of Database Systems (PODS), pp. 90-101, June 2005;

[22] P. Th. Eugster, P. A. Felber, R. Guerraoui and A.-M. Kermarrec, “The many Faces of Publish/subscribe”, ACM Computing Surveys (CSUR), volume 35, issue 2, pg: 114-131, June 2003;

[23] L. Aldred, W. M. P. van der Aalst, M. Dumas and A. H. M. ter Hofstede, “On the Notion of Coupling in Communication Middleware”, On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, Lecture Notes in Computer Science, volume 3761/2005, pg. 1015-1033, 2005;

[24] T. Schlossnagle, “Scalable Internet Architectures”, Sams, 2006;

[25] S. Tarkoma and K. Raatikainen, “State of art review of distributed event systems”, Minema Events Report C0-04, February 2006;

[26] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang, “A Case for End System Multicast”, IEEE Journal on Selected Areas in Communications (JSAC), volume 20, issue 8, pg: 1456-1471, October 2002;

[27] M. Castro, P. Druschel, A.-M. Kermarrec and A. Rowstron, “SCRIBE: A Large-scale and Decentralized Application-level Multicast Infrastructure”, IEEE Journal on Selected Areas in Communication (JSAC), volume 20 issue 8 pages 100-110, 2002;

[28] A. Carzaniga, D.S. Rosenblum, and A.L. Wolf, “Design and Evaluation of a Wide-Area Event Notification Service”, ACM Transactions on Computer Systems, volume 19, issue 3, pg: 332-383, August 2001;

[29] C. Diot, B.N. Levine, B. Lyles, H. Kassan, and D. Balendiefen, “Deployment numbers for the IP Multicast services and architecture”, IEEE Networks - Special number Multicasting, volume 14, issue 1, pg: 78-88, 2000;

[30] G. Cugola and G. P. Picco, “REDS: a Reconfigurable Dispatching System”, Proceedings of the 6th international workshop on Software engineering and middleware, pg. 9-16, 2006;

[31] M.A. Jaeger, G. Mühl, M. Werner, H. Parzyjegla and H.-U. Heiss, “Algorithms for Reconfiguring Self-Stabilizing Publish/Subscribe Systems”, Autonomous Systems – Self-Organization, Management, and Control, Springer, pg: 135-147, September 2008;

[32] C. Esposito, D. Cotroneo and A. Gokhale, “Reliable Publish/Subscribe Middleware for Time-sensitive Internet-scale Applications”, Proceeding of the 3rd ACM International Conference on Distributed Event-Based Systems (DEBS), July 2009;

[33] P. Costa, M. Migliavacca, G.P. Picco and G. Cugola, “Epidemic Algorithms for Reliable Content-Based Publish-Subscribe: An Evaluation”, Proceeding of the 24th IEEE

International Conference on Distributed Computing Systems (ICDCS'04), pg: 552-561, March 2004;

[34] L. Rizzo and L. Vicisano, “RMDP: an FEC-based reliable multicast protocol for wireless environments”, ACM SIGMOBILE Mobile Computing and Communications Review (MC2R), volume 2, issue 2, pg: 23-31, April 1998;

[35] D. Garlan, S. Cheng, A. Huang, B. Schmerl and P. Steenkiste, “Rainbow: Architecture-Based Self-Adaptation with Reusable Infrastructure”, IEEE Computer, Vol. 37, No. 10, Pg. 46-54, 2004;

[36] B. H. Cheng, R. Lemos, H. Giese, P. Inverardi and J. Magee, “Software Engineering for Self-Adaptive Systems”, Software Engineering for Self-Adaptive Systems, Lecture Notes In Computer Science; Vol. 5525, 2009

[37] C. Esposito, D. Cotroneo and S. Russo, “An Investigation on Flexible Communications in Publish/Subscribe Services”, Proceedings of the 8th IFIP Workshop on Software Technologies for Future Embedded and Ubiquitous Systems (SEUS 2010), October 2010;

[38] Object Management Group, “Data Distribution Service for Real-time Systems (DDS)”, version 1.2, 2007;

[39] S. Rafaeli and D. Hutchison, “A survey of key management for secure group communication”, ACM Computing Surveys (CSUR), Vol. 35, No. 3, Pg. 309-329, September 2003;

[40] M. Srivatsa and L. Liu, “Secure Event Dissemination in Publish-Subscribe Networks”, Proceedings of the 27th International Conference on Distributed Computing Systems (ICDCS 2007), Pg. 22-29, 2007;

[41] L. I. W. Pesonen, D. M. Eyers and J. Bacon, “A capabilities-based access control architecture for multi-domain publish/subscribe systems”, Proceedings of the Symposium on Applications and the Internet (SAINT 2006), Pg. 222–228, January 2006.

Copyright Disclaimer

Copyright reserved by the author(s).

This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/3.0/).