A Multi-core Context-Aware Management Architecture for Mixed-Criticality Smart Building Applications

A. C. Dimopoulos, G. Bravos, G. Dimitrakopoulos, M. Nikolaidou, V. Nikolopoulos, and D. Anagnostopoulos
Harokopion University of Athens
Department of Informatics and Telematics
Athens, Greece
Email: {alexdem,gbravos,gdimitra,mara,basil,dimosthe}@hua.gr

***Abstract*—The Future Internet era dictates the deployment of high complexity systems having a variety of properties such as adaptivity, self-configuration and self-optimization. This paper discusses on the design and deployment of a context-aware management architecture, for mixed-criticality applications. The architecture of an autonomous smart building system is described, focusing on diverse requirement satisfaction, including response time, energy efficiency and users' needs. The main objective of the architecture is to investigate the relationships between context awareness characteristics and mixed-criticality aspects, and to identify ways to exploit the one in favor of the other. The analysis provides solid evidence that such an architecture is realistic and can lead to highly competitive systems.**

I. Introduction

The Internet of Things (IoT) connects embedded devices, sensors and actuators to the Internet. Estimations show that around 50 billion devices will be connected to the Internet in 2020 [1]. Novel applications, ranging from autonomous driving and robotics [2], [3] to healthcare, which improve the quality of human life, will be enabled by IoT. Common to these applications is the collection of sensor information, the control of actuators as well as time-critical communication and control tasks. The gathering and processing of data acquired by those devices will result in an increase of adaptive data aggregation devices. Thus, the architecture for such systems must be designed in an organized, scalable, and efficient way that leads to a seamless deployment of IoT applications on a generic platform.

Several challenges rise based on these new requirements. For instance, the "silo effect" problem, where different applications do not share anything related to the infrastructure, is commonly present in the big variety of applications developed based on IoT technologies. The same issue may be found in the lack of global standardized methods used to collect, send and process data from sensors and actuators [4]. This issue is far more important in the cases that mixed-criticality applications shall be handled by a management system rapidly and effectively.

Moreover, IoT communication systems need to support real-time low-latency operating modes. These properties cannot be realized by a distant cloud connection due to the reliability of network communications as well as the finite speed of light. Hence a network of autonomous nodes near to the IoT devices is required to enable low-latency context aware operation, which could be valid for mixed-criticality applications.

Another critical issue is the usage of different network access technologies. There is not a unique solution or standard to enable a universal platform between the nodes and the servers. Instead, each vendor decides which technology fits best in their architecture to implement it. Hence, current infrastructure has to deploy specific and proprietary solutions and protocols to be able to connect and interact with things. Another aspect of these protocols is the unencrypted communication between the different parties; anyone with access to the wireless medium can read or intercept any message. Therefore, a solution that offers dynamic reconfiguration secure capabilities and more independence from the hardware platform is required.

Last, in the near future, embedded multi-core applications will integrate multiple functionalities on a single chip. These applications are expected to show different criticality levels which could lead to a significant and potentially unacceptable increase of engineering and certification costs. Several approaches to keep these costs on a reasonable level follow the idea of partitioning in time and space and are already in the research pipeline. In order to develop reasonable approaches that allow the integration of different types of mixed-criticality applications on a single system, potential applications need to be examined.

In the light of the above, the goal of this paper is to describe an architecture and define potential applications for the management of mixed-criticality smart building applications. The proposed architecture (i) responds to the silo effect problem, through organizing the architecture with "intelligent agents", enabling mixed-criticality applications to share resources provided by the infrastructure and the underlying network access technologies; (ii) targets autonomy at many levels enabling low latency context-aware operation, through the utilization of autonomous decision making modules that provide context-aware solutions in terms of distributing the resources on a real-time basis among mixed-criticality applications and (iii) is capable of keeping engineering and certification costs at low

level, following the idea of partitioning in time and space.

II. Motivation

A. Context aware IoT systems

Context-Awareness is an area of Computer Science which deals with the adaptation of computing systems to the user's current context. The most recognized definition for the term "context" was from Day [5]: "*Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.*" The key-concepts related to context awareness include (i) context acquisition, which is related to the gathering of the data required for context awareness, and can be categorized based on several sets of criteria [1], [6], [7], (ii) context reasoning, which is related to the understanding of the data towards context awareness [8], (iii) knowledge-based decision making, which is related to the exploitation of context awareness [9], [10], and (iv) context delivery, which is related to methods available for delivering context to consumers [11].

On the other hand, regarding mixed-criticality systems (MCS), they were first introduced by Vestal (of Honeywell Aerospace) in 2007 [12]. It employed a somewhat restrictive work-flow model, focused on a single processor and made use of Response Time Analysis [13]. It showed that neither rate monotonic nor deadline monotonic priority assignment was optimal for MCS; however the optimal priority assignment algorithm presented in [14] was found to be applicable.

Baruah and Vestal [15] generalized Vestal's model by using a sporadic task model and by assessing fixed job-priority scheduling and dynamic priority scheduling. They demonstrated the important result that EDF (Earliest Deadline First) does not dominate fixed priorities when criticality levels are introduced, and that there are feasible systems that cannot be scheduled by EDF. The research effort of Huber et al. [16] addresses multi-processor issues and virtualisation. It focused on AUTOSAR and resource management (encapsulation and monitoring) with time-triggered applications and a trusted network layer. Further impetus to defining MCS as a distinct research topic came from the white paper produced by Barhorst et al. [17], and a workshop report from the European Commission [18]. A mixed-critical system can be defined as "an integrated suite of hardware, operating system and middleware services and application software that supports the execution of safety-critical, mission-critical, and non-critical software within a single, secure compute platform" [19]. In order to fulfill these requirements a strong isolation of applications is needed. Such an isolation is succeeded if the execution of one application is not influenced by the behavior of other applications. According to recent research efforts, such isolation may be either spatial, i.e applications must be executed in independent memory address spaces or temporal, where the real-time behavior of an application must be correct independently of the execution of other applications [20].

Based on the technology advances in both these research fields, this paper focuses on bringing them together, motivated by the need to provide answers to the following questions: *How is context awareness related to multi criticality in systems? To what extend can the data exploited in the framework of context awareness, be used in order to define and distinguish between critical and non-critical applications? How can multi critical applications affect the context of a system?*

We will try to provide answers to the aforementioned questions, through the presentation of an envisioned context-aware smart building management system with multi-critical applications, as described in what follows.

B. Context acquisition in mixed-criticality systems

Let it be noted that in the context of this paper, sensors are considered as the primary source of any type of information that leads to contextual acquisition. Context acquisition can be categorized according to a set of criteria, such as: (i) Initiation: sensor data can be acquired either after a request made from a software component to the sensors (this method is called a "pull" method), or after the sensors periodically send information to the software component ("push" method) [6]; (ii) Frequency: sensor data can be acquired either in an instant mode, whenever a specific value threshold is violated, or in a periodic interval mode, through transmitting data every X time units. In both types, either push or pull methods can be utilized [1]; (iii) Source: context related data can be acquired either directly from sensor hardware, or indirectly from a middleware employed for this purpose, or even from context servers through web service calls [7]; (iv) Sensor type: context data is derived either from physical sensors that generate data, or from virtual sensors that retrieve data from multiple sources and combine it to generate sensor data, or even from logical sensors that combine physical and virtual sensors to provide meaningful information [21]; and (v) Acquisition process: the process to acquire data requires either sensing data, deriving data through calculations, or manually providing data in terms of user preferences [22].

Such data acquisition processes may be highly affected by different criticality levels of applications within the same system. For instance, a categorization can be carried out based on a new criterion, introduced by the criticality level of each application where a data set may refer to.

C. Mixed-criticality systems and Context reasoning

Context reasoning can be achieved with many different techniques, most of which derive historically from the fields of AI and machine learning. These various context reasoning models can be divided into different categories, depending on their principles. There are those that follow a supervised learning technique and those that follow an unsupervised one, i.e. the distinction is whether or not a training set is used to calibrate the reasoning model. Another category is whether the reasoning is inferred using rules or logic. An in-the-middle solution between logic and rule based approaches can

be achieved by using ontology semantics, especially now that semantic web [8] is evolving rapidly.

Regarding the proposed system, the computational resources are not unlimited but to the contrary very limited, especially on each core. Hence, although the computational power is increased by arranging the multiple cores in a distributed system form, it is considered as the most reasonable solution to keep the computational cost as low as possible. Therefore, given the relatively small number of produced rules that can describe the decision making of the proposed system, it is simpler and faster to follow the path of rule based reasoning. Furthermore, in order to exploit the distributed nature of the multiple cores, the rules can be divided into specific domains, and be distributed along the multiple cores. Each such domain, should be as autonomous as possible from all the other domains; especially all the low level rules should be independent and only at higher level should be conflicts between the domains. In this fashion, the computational burden is distributed along the cores and simple decisions can be made by each core, depending on the input. Only for the few decisions that need to be combined with decisions from other domains, an extra communication cost will be added to the total computation cost.

D. Context-aware and knowledge-based decision making

The unquestionable deployment of high complexity systems in modern ICT can be facilitated through several concepts, one of which is the reconfigurability concept, often seen as an evolution of "software defined radio" [23]. Moving one step further, complexity can be fought through the design of communication infrastructures on the premises of self-management and learning, i.e. acting in a context-aware manner and also exploiting cognitive networking principles [10]. In general, a cognitive system is capable of retaining knowledge from past interactions with the external environment and deciding upon its future behavior based (i) on this knowledge, (ii) other goals and also (iii) policies, so as to optimize its performance. It is anticipated that cognitive systems can facilitate the design, development and integration of novel services and applications, needed to support diverse requirements in terms of Quality of Service (QoS) or Quality of Experience (QoE) [9]. Cognition as a concept might involve various parts of a communication infrastructure, i.e. network segments, terminals, e.t.c. and can be applied either in a centralized or in a distributed manner. However, it is believed that distributed, even autonomic solutions might be more appropriate in terms of complexity. Thus, research is moving towards the adoption of solutions that lie in accordance with the context-aware decision making paradigm [7]. Nevertheless, context awareness has rarely been seen until now in relation with mixed- criticality applications.

As already stated, mixed-criticality systems are characterized by the integration of critical and non-critical applications on the same computing platform. In order to fulfill these requirements a strong isolation of applications (critical and non-critical) is needed. An application is isolated from others if its execution is not influenced by the behavior of the other applications. Nevertheless, every process has implications to the overall system. Scheduling policies and scheduling techniques to achieve the independent certification of critical partitions of applications have to be consolidated. The scheduling problem is one of the most active research area, resulting in proposals of different approaches to deal with partitioned systems. However, techniques that can guarantee the incremental scheduling of partitions are still needed.

Using intelligent agents along with distributed context-aware and knowledge-based decision making algorithms, can result in such scheduling issues to be carried out in a more efficient way. Context awareness can be the key towards enhanced multi - criticality scheduling, through the exploitation of the capabilities of reconfigurable / self-adapting infrastructures.

Another critical issue where the proposed approach aims to provide advances with respect to state of the art, is the proposition of a fully reconfigurable platform, being capable of dynamically, transparently and securely utilizing the most appropriate network access technology for operation, thus defining and isolating applications if necessary. Shared hardware resources in multicore systems have need stated to introduce several challenges, as they have an impact on temporal isolation. The use of shared resources (such as L2/L3 cache, memory, bus, IO, e.t.c.) by partitions running on different cores in parallel introduces an interference in the overall execution. This interference impacts directly in the independent behavior of partitions and, as consequence, in its independent certification.This challenge though can be addressed through the transparent and secure utilization of the most appropriate network access technology for operation, by avoiding or minimizing interferences.

Finally, the proposed system targets autonomy at many levels enabling low latency context-aware operation, via the utilization of autonomous decision making modules that provide context-aware solutions in terms of distributing the resources on a real-time basis among mixed-criticality applications.

III. Multi-Core Context-Aware Architecture

Our envisioned architectural framework is depicted in Figure 1. It must be noted that our architecture supports plug and play connectivity and is designed to be scalable, self-optimized, and multi-layered.

Starting from the lowest layer, the sensing units are found which report to an authority (aggregator) that is responsible for providing web based RESTful services. As it has already been mentioned due to the "silo effect" issue, each different sensor most likely uses different technology for communicating, and hence for each different technology one different aggregator is assigned. Therefore, each set of same technology sensors is responsible for reporting its own data to a designated data aggregator. In Figure 1, two different sensor technologies are depicted, the MICAZ and IMA sensors that each communicate with two different aggregators, i.e. a Raspberry Pi and an IMA aggregator respectively.

At the next layer, that of the aggregators, various different procedures are serviced, e.g. keeping an up-to-date list of

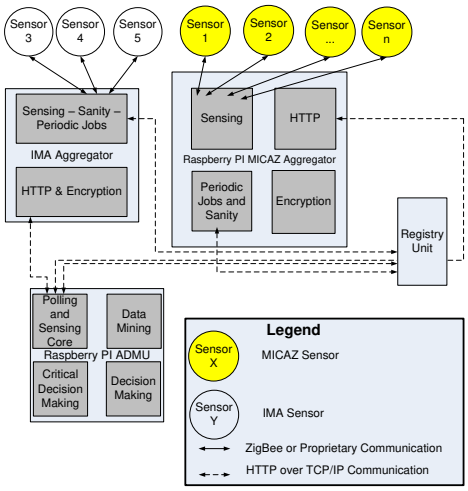

Fig. 1. Overview of the architecture

available sensors, making sensors' data available through the RESTful services and communicating with the sensors. The only limitation is that each aggregator unit must provide RESTful services in the standard way. A very important aspect of the communication is the secure transmission of data from the aggregators to the next layer. It must be clarified that the encryption concerns the communication between the aggregators and the next layer, where aggregated data are transmitted; the communication between the sensors and the aggregators is unencrypted since the sensor cannot tackle the computational burden of encryption. Even if the sensors were computationally capable of encrypting data, the communication between the sensors and the aggregator is usually over a proprietary communication protocol and hence cannot be changed by a third party.

One layer above the aggregators is the middleware of the Registry Unit. Since there are many aggregators with different IP addresses, which all employ the TCP/IP stack to provide their RESTful services over HTTP, one should be able to discover available aggregators without much trade-off. Hence, that need is covered by the Registry Unit. The registry unit is responsible for keeping an up to date list of the services available from the aggregators and is contacted by the aggregators in order to register their services. Moreover the aggregators can update and delete any service they own. Additionally, the registry unit is responsible for polling said registered services to check on their availability and if not remove them from its list.

The top layer is that of the autonomic decision making unit (ADMU). This component adds the required intelligence on the proposed system, by deciding on actions to be carried out from smart agents, through their services. The decisions are based on policies set by third parties, like users or energy efficiency policies, by taking into consideration user feedback/preferences, the context information from the sensing units and information from the pattern repository. Alongside the aggregators, the ADMU is responsible for the self-healing and self-optimization of the system.

Based on the above description, it is clear that in the case of either the aggregator or the ADMU, the system must run more than one task, namely sensing, sanity checks, and encrypted communication with the higher layer for the aggregator and polling/sensing, data mining, encrypted communication with the lower layer, and decision making for the ADMU. All the above tasks and especially the encryption/decryption phase of the communication are so computationally intense for a simple uniprocessor embedded system that would definitely lead to lags and latencies. Luckily, nowadays there are many options for multi-core embedded systems, which allow the smooth concurrent execution of all tasks and hence, both the aggregator and the ADMU of the proposed system are implemented on multi-core processor systems.

A. *The registry unit*

As described previously, the registry unit constitutes a simplified DNS-like service (Figure 2). It supports three kinds of requests: to GET data, to POST/REGISTER data, and to DELETE data. A custom made http server hosts the services consisting the Registry API.

Any component may poll the registry unit so as to get a list of available components (either aggregators or application/service components) in the area. Moreover, it may ask for available services of a specific component. A registry entry consists of the component description, information for the IP it is available at, and the components services followed by a short description of each service. For example, an application component may poll the registry with an aggregator id and a sensor id as HTTP parameters. The registry should let the application component know about the available services the aggregator offers on sensor level on that specific sensor. Such services might be for instance, a switch toggle, a specific sensor reading or a battery level. Components are the only ones responsible for posting the data to the registry unit through a registration procedure. This procedure is done over HTTP as well. A component is also responsible for renewing its registry

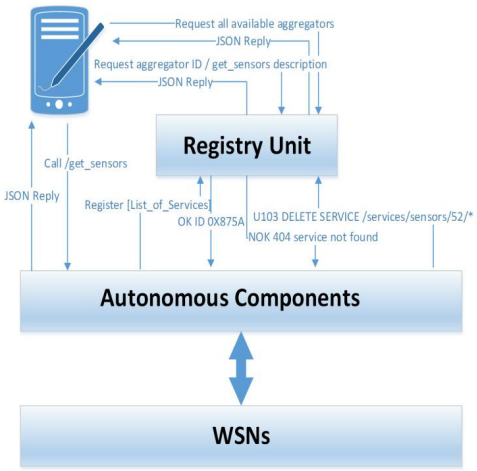

Fig. 2. The registry unit

records when a service/sensor node is no longer available. The registry unit can opt to delete an component and the services it offers, denoted with the unique ID, assigned to at the register process, if the component has not renewed his records in a while. The services provided are currently fully RESTfull and therefore can be easily manipulated and be used. Moreover all reply messages are in JSON form to allow for easier manipulation of the data received. A developer willing to develop an application using a wireless sensor network (WSN) that uses the registry unit, can simply poll the registry for the services' descriptions and get started effortlessly.

The registry unit acts as discovery service for the users currently entering the smart building and a registry service for the autonomous components currently available. As such, no information about the underlying WSN and sensor types is maintained in the registry. Aggregators are responsible for the correct documentation of their services and that allows us to consider a layer of abstraction over the WSN.

B. The ADMU Unit

The main idea behind ADMU is the development of the system-level schemes fostering autonomous functionalities as the technological solutions for mixed-criticality problems. In particular, it is anticipated that the existence of centralized control entities cannot be guaranteed in mixed-criticality scenarios, rendering autonomous functionalities a necessity. The solutions are based on the autonomic networking concept, utilizing the newest results from several research areas, including dynamic distributed systems, self-management, self-healing, self-organizing and other self-x related management concepts, artificial intelligence, and machine learning.

In particular, having as inputs: 1) contextual information (number of sensors, types of sensors, types of services, criticality levels), 2) profile information (capabilities of sensors, user preferences, service profiles), as well as 3) policies information (restrictions to the solution space invoked by the user or a regulator) autonomic decision making algorithms will be able to decide on prioritizing a service against another service, or changing the parameters of a service provided, so as to provide the user with the maximum QoS levels possible, whilst satisfying a set of pre-defined criteria.

Decision making is based on a combination of supervised learning based on known regulations and policies while taking into account the inputs provided from the sensors and the registry unit, as well as parameters learned and/or configured based on accumulated history of actions.

IV. Usage examples

Having implemented the above architecture, any user who wishes to find a service can poll the registry unit for its services list and call the service available at the aggregators IP with the pattern that is described in the record of the registry unit. It should be noted that when we refer to users, we don't necessarily limit to human beings, but applications, smart agents e.t.c. can be included as well. All of the above communications are described in Figure 2 extensively.

Moreover, agents that provide policies and patterns could be included at the top layer. As far as policies are concerned, looking strictly from an energy efficiency spectrum, a smart meter is able to provide data about energy consumption in our system, i.e. a smart-home, and provide this information to the ADMU, that in turn should then decide on how to exploit these data. On the other hand a user, i.e. an inhabitant of the said smart-home, from his/her actions provides patterns of usage that are compiled in the pattern repository. The pattern repository collects data from the sensors, the smart devices, and operating sets and procures patterns of usage, which are used from the ADMU to make decisions. These decisions clearly follow the multi-critical nature of our architecture and at different times or under different context, the same actions may have different criticalities.

Two elementary case studies will follow, describing everyday usage examples of the system. Additionally, the mixed-criticality nature of the proposed architecture is demonstrated especially in the second example.

A. Case I : Energy efficient heating

Let us assume a system implementation that is able to measure real-time energy consumption. The proposed system aims to (i) act as a pervasive system in order to satisfy the user's needs and (ii) minimize the overall energy consumption in the house. This leads to a trade-off that the system has to handle. Based on the context, the system knows that the temperature in a specific room has to be at least 20 degrees Celsius after 15.00. This knowledge can either be given by the user e.g. through a mobile application or it can be inferred automatically based on previous user's actions, e.g. the user keeps setting that temperature to 20 degrees for the last 3 days at the same time. However, for some reason the user needs have changed and the desired temperature must be reached two hours earlier. According to the system's analysis regarding energy consumption, the optimum operation of the heating system would lead in the room having less than 20 degrees at that specific time of the day, since it is more energy efficient to gradually increase the temperature over time than to increase it sharply. Nevertheless, the critical service/requirement is the one regarding the minimum temperature, so the system should be able to decide based on that, regardless the effects on the energy consumption.

B. Case II: Prevent further damages from water leakage

Let us assume a system implementation that is able to sense humidity, i.e. humidity sensors are connected to the aggregators. The proposed system aims to act as a deterrent system against water leakage and should take specific actions in such cases regardless of any other direction/guideline/context from the user. When a sensor measurement indicates very high humidity above a threshold or even leakage for example in the boiler, the system instantly shuts down the electric power to the boiler and any adjacent electrical appliances. Furthermore, the user should be notified and if possible the building janitor as well. Ideally, the system should also have access to an electric

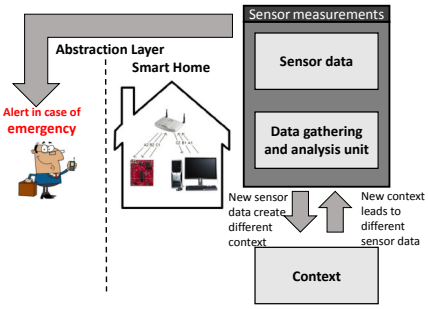

Fig. 3. Alert in case of emergency

valve to stop the main water supply, as depicted in Figure 3. Thus, all the above actions are set to the most (safety) critical level and must be satisfied before all others. Obviously any other decisions that are possible contractive to the most critical actions will be ignored. It goes without saying that under other circumstances, i.e. if the humidity levels are high but not implying a water leakage, other decisions are made (e.g. ignition of the ventilation system) classified as non-critical.

Apart from the above two indicative examples, the final fully evolved system will be tested against various situations. As already mentioned, many different sensors will be included and hence many rules of different nature and different criticality will be given. The dynamic nature of the system itself will allow it to be trained according to each user's needs and demands.

V. Anticipated Extensions and Conclusions

This paper has presented a context-aware, smart building management multi-core architecture, for tackling mixed-criticality applications in an IoT context. The current status of the envisioned architectural framework is based on a basic number of different sensors as well as on a registry unit with basic functionalities. Future enhancement of the system is expected to take place in the near future, using and integrating Freescale ARM based wireless transceivers and microcontrollers in the 802.15.4 2.4GHz band.

In addition to that, more sensors are to be embedded into the PCB or to be designed as add-ons to the existing platform including RFID/NFC, Fingerprint readers, Gyroscopes / Accelerometers / Magnetometers as well as Bluetooth sensors to be used for the users' profile identification.

Regarding the data aggregation and the registry unit, this will be enhanced using an embedded module based on the Freescale Vybrid platforms with a Cortex A5 + M4 that will also be developed and allow the smart building's autonomic system of mixed-critically solutions. For instance, the Cortex A5 will enhance the system as it enables to include a Linux platform to port the current existing platform, while the M4 enables including a barebone implementation or the MQX RTOS to support critical real-time processes.

Furthermore, autonomous, knowledge-based decision making units will be deployed for tackling mixed-criticality related decisions and for optimizing the performance of the proposed architecture, considering a number of criteria (policies). Finally, communication interfaces will also be added to the system, namely Ethernet with a fully TCP/IP stack, 802.15.4 transceiver and antenna e.t.c..

Acknowledgement

The research leading to these results has received funding from the ARTEMIS Joint Undertaking under grant agreement n^o 621429.

References

[1] D. Evans, "The internet of things: How the next evolution of the internet is changing everything," *CISCO white paper*, vol. 1, p. 14, 2011.
[2] M. Essers and T. Vaneker, "Evaluating a data distribution service system for dynamic manufacturing environments: A case study," *Procedia Technology*, vol. 15, pp. 621 – 630, 2014.
[3] ——, "Evaluating a prototype approach to validating a dds-based system architecture for automated manufacturing environments," *Procedia CIRP*, vol. 25, pp. 385 – 392, 2014.
[4] O. Logvinov, "Open standards will enable the IoT growth," 2014.
[5] A. K. Dey, "Understanding and using context," *Personal and ubiquitous computing*, vol. 5, no. 1, pp. 4–7, 2001.
[6] M. Baldauf, S. Dustdar, and F. Rosenberg, "A survey on context-aware systems," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 2, no. 4, pp. 263–277, 2007.
[7] H. Chen, T. Finin, A. Joshi, L. Kagal, F. Perich, and D. Chakraborty, "Intelligent agents meet the semantic web in smart spaces," *Internet Computing, IEEE*, vol. 8, no. 6, pp. 69–79, 2004.
[8] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific american*, vol. 284, no. 5, pp. 28–37, 2001.
[9] "Project "End-to-End Efficiency" (E3)," 7th Framework Programme (FP7) of the European Commission, Information and Communication Technologies (ICT). [Online]. Available: www.ict-e3.eu
[10] R. W. Thomas, D. H. Friend, L. DaSilva, and A. B. Mackenzie, "Cognitive networks: adaptation and learning to achieve end-to-end performance objectives," *Communications Magazine, IEEE*, vol. 44, no. 12, pp. 51–57, 2006.
[11] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 1, pp. 414–454, 2014.
[12] S. Vestal, "Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance," in *28th Real-Time Systems Symposium (RTSS)*. IEEE, 2007, pp. 239–243.
[13] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. J. Wellings, "Applying new scheduling theory to static priority pre-emptive scheduling," *Software Engineering Journal*, vol. 8, no. 5, pp. 284–292, 1993.
[14] N. C. Audsley, "On priority assignment in fixed priority scheduling," *Information Processing Letters*, vol. 79, no. 1, pp. 39–44, 2001.
[15] S. Baruah and S. Vestal, "Schedulability analysis of sporadic tasks with multiple criticality specifications," in *Real-Time Systems, 2008. ECRTS'08. Euromicro Conference on*. IEEE, 2008, pp. 147–155.
[16] B. Huber, C. El Salloum, and R. Obermaisser, "A resource management framework for mixed-criticality embedded systems," in *34th Industrial Electronics (IECON)*. IEEE, 2008, pp. 2425–2431.
[17] J. Barhorst, T. Belote, P. Binns, J. Hoffman, J. Paunicka, P. Sarathy, J. Stanfill, D. Stuart, and R. Urzi, "White paper: A research agenda for mixed-criticality systems," in *CPS Week 2009 Workshop on Mixed Criticality: Roadmap to Evolving UAV Certification*, 2009.
[18] H. Thompson, "Mixed criticality systems," 2012.
[19] S. Baruah, H. Li, and L. Stougie, "Towards the design of certifiable mixed-criticality systems," in *16th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2010, pp. 13–22.
[20] A. Burns and R. Davis, "Mixed criticality systems-a review," *Department of Computer Science, University of York, Tech. Rep*, 2013.
[21] A. Schmidt and K. Van Laerhoven, "How to build smart appliances?" *Personal Communications, IEEE*, vol. 8, no. 4, pp. 66–71, 2001.
[22] A. A. Alidin and F. Crestani, "Context acquisition in just-in-time mobile information retrieval," in *Information Retrieval & Knowledge Management, International Conference on*. IEEE, 2012, pp. 203–207.
[23] W. Hasselbring and R. Reussner, "Toward trustworthy software systems," *Computer*, vol. 39, no. 4, pp. 91–92, 2006.