2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing

Self-Organised Middleware Architecture for the Internet-of-Things

Pedro Maló, Bruno Almeida, Raquel Melo
Centro de Tecnologia e Sistemas,
Departamento de Engenharia Electrotécnica,
Faculdade de Ciências e Tecnologia,
Universidade Nova de Lisboa
Caparica, Portugal
pmm@uninova.pt, bma@uninova.pt,
raquelabrantesmelo@gmail.com

Kostas Kalaboukas
SingularLogic, Athens, Greece
kkalaboukas@singularlogic.eu

Philippe Cousin
Easy Global Market, Paris, France
philippe.cousin@eglobalmark.com

***Abstract*— Presently, middleware technologies abound for the Internet-of-Things (IoT), directed at hiding the complexity of underlying technologies and easing the use and management of IoT resources. The middleware solutions of today are capable technologies, which provide much advanced services and that are built using superior architectural models; they however fail short in some important aspects: existing middleware do not properly activate the link between diverse applications with much different monitoring purposes and many disparate sensing networks that are of heterogeneous nature and geographically dispersed. Then, current middleware are unfit to provide some system-wide global arrangement (intelligence, routing, data delivery) emerging from the behaviors of the constituent nodes, rather than from the coordination of single elements, i.e. self-organization. This paper presents the SIMPLE self-organized and intelligent middleware platform. SIMPLE middleware innovates from current state-of-research exactly by exhibiting self-organization properties, a focus on data-dissemination using multi-level subscriptions processing and a tiered networking approach able to cope with many disparate, widespread and heterogeneous sensing networks (e.g. WSN). In this way, the SIMLE middleware is provided as robust zero-configuration technology, with no central dependable system, immune to failures, and able to efficiently deliver the right data at the right time, to needing applications.**

***Index Terms*— Self-organization, Multi-layer, Middleware, WSN, Internet of Things, IoT, ARTEMIS, SIMPLE, PROBE-IT.**

I. INTRODUCTION

While the exact definition of self-organization depends of application field and differs among individuals, it is safe to say that a self-organized system consists of different elements, whose global arrangement emerges from the behaviors of its elements rather than from the coordination of a single element. Notably, self-organization is not a functionality of a system, but a property of itself.

Self-organization can be found in many types of systems, in many different research fields, but perhaps the most illustrative can be found in biology. Flocks of birds and schools of fish can be found migrating thousands of miles every year and creating dizzying spectacles evading predators, without central coordinator.

To obtain the self-organization it is usually required that the elements of the system are able to interact with their surrounding environment in some kind of way. This includes some form of interaction with the other elements in the system, being unnecessary to interact with all of them, and keeping the individual element agnostic of the presence of other elements when not interacting.

Wireless Sensor Networks [4] consist of many nodes that have very strict limitations in terms of computational power, memory and data storage capacity. Additionally, they typically have a limited supply of energy with which they will have to stay operational as long as possible. Finally, their method of communication, using cheap radios with little output power, is prone to failures, causing lost packets, collisions, in unpredictable forms. From this collection of unreliable nodes, a reliable network must be constructed, capable of doing sensing tasks that applications' need [3].

On a network level the use of self-organization has, by now, been acknowledged. In the absence of maestro nodes that have the capacity of continuously coordinate and organize the actions of other many nodes in a changing environment, not many viable alternatives exist in order of maintaining the network reliable. One of the other options nullifies the ad-hoc nature of a wireless technology, giving the control to a central node becoming dangerously vulnerable to changes in the environment.

Thus, there is a clear need for self-organizing and highly distributed algorithms on the MAC and routing layers, that allow nodes to evaluate their situation and modify their behavior locally. This way, the network as a whole can adapt to a changing environment and deal with the inherent unreliability of individual nodes.

While much effort has been put in the development of such distributed network protocols, little attention has been paid to the need for self-organization on the application level. Developing an application for a Wireless Sensor Network is not trivial, where an application is specified on a high level, and then compiled to be executable for individual nodes. Leaving the processing at the application level generates static applications that are highly dependent on the topology of the network, which must thus be known in advance.

978-0-7695-5046-6/13 $26.00 © 2013 IEEE
DOI 10.1109/GreenCom-iThings-CPSCom.2013.92

II. SIMPLE SELF-ORGANIZED MIDDLEWARE

The SIMPLE Self-Organized Middleware [1,2] defines a multi-layer approach able of withstand the difference of technologies and methods in-between the several parts of a Wireless Sensor Network (WSN) system. This is achieved through the definition of one API and two layers, which will be then thoroughly explained throughout this document:

- The SIMPLE API: Is used by applications to access the SIMPLE sensor network, and is able of translating application queries to a language understood by WSN;
- The Data Distribution Middleware Layer: Joins the gap between the sensing devices and the application. Its main feature is to deliver data, and to be an enabler to lower the processing power needed at the WSN devices;
- The WSN Layer: Describes how self-organized network works, such as the subscription processing methods and the node components. It also includes the specification of the implementation over MyriaChap [6], which is a component-based Wireless Sensor Network platform.

Being highly dynamic implies that not only the way the system is organized has to be different, also the way that the interactions with the system occurs changes. This high dynamicity leads to that the devices performing the monitoring of places are not known to the application, which involves the change of the approach to these systems.

The SIMPLE approach relies in a combination of a subscription-based system that uses spatial location of information sources for monitoring specific places following successful works on scalability and efficiency approaches for WSNs using geographic information [10].

In the SIMPLE Self-Organized Middleware when a requester wants to subscribe information, it sends a subscription to the spatial area in which it wants to gather responses in opposition of sending it directly to a provider.

As an example, if the requester wants to receive information about alterations in the temperature of a specific room, it sends a subscription to the coordinates of the room. In this approach it is indifferent if the response is provided by device A, B or C. This is possible since sensor devices know the coordinates in which they are located, and their sensing range in which they are able of gathering requests. The range is defined taking in concern the Cartesian equation of the circle:

$$(x_{provider} - x_{subscription})^2 + (y_{provider} - y_{subscription})^2 \leq r_{provider}^2$$

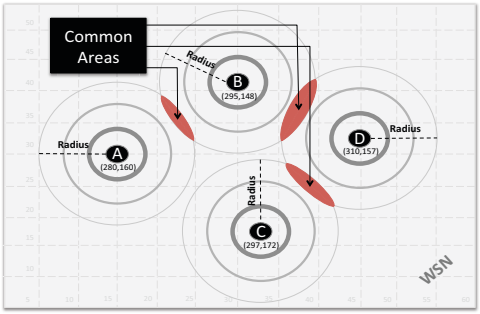

Figure 1. Spatial Localisation Area Definition

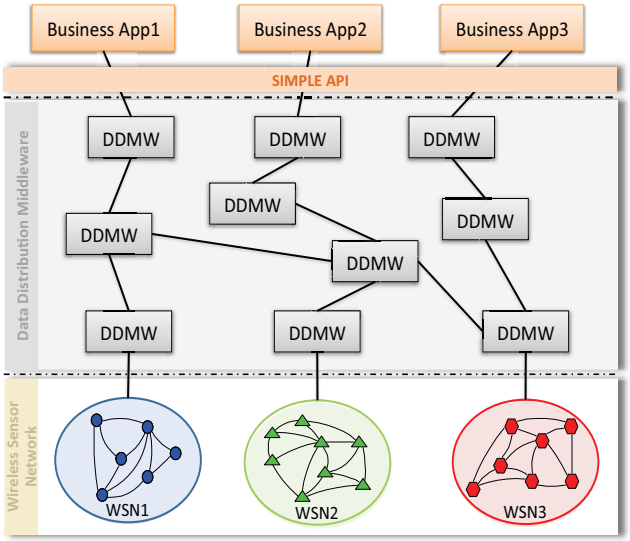

Figure 2. SIMPLE Multi-layered Approach.

III. MULTI-LAYER ARCHITECTURE

A SIMPLE deployment consists of a large number of WSNs established at different locations where data collected by sensor nodes is required for processing in different steps of logistic (monitoring of cargo), domestic (controlling inventory) or manufacturing systems (tracking parts).

Such a system could quickly become unmanageable without suitable organization and hierarchy. Approaches such as Global Sensor Network (GSN) [11] facilitate sensor node grouping/handling, but fails on long-distance networks, with peers that are not always connected and are mobile.

The hereby-suggested multi-layer approach introduces a middleware layer in between applications and WSNs that remains reliable in highly dynamic large-scale widespread distributed environments. The proposed approach is inspired on past works on data dissemination models for WSNs [5].

A. Layers Organization

The SIMPLE system is divided into one API - SIMPLE API - and two layers - Data Distribution Middleware and Wireless Sensor Networks - as depicted in the Figure 2.

1) SIMPLE Application Programming Interface

This API represents the integration needed at the application side to connect to the SIMPLE system. This is the entry point that is used by the business application to access the sensor network.

The SIMPLE API uses a domain specific language, the SIMPLE Query Language, to communicate with the applications. By default, requests can be made through web services using the SOAP protocol to access one of the middleware servers.

Additionally, the SIMPLE API doesn't run on a unique central point, but it runs on every instance of the middleware, so each application can instantiate a new middleware at their location and expand the distributed network range, making the application part of it.

2) SIMPLE Data Distribution MiddleWare (DDMW)

The Data Distribution Middleware represents the part of the SIMPLE self-organized middleware that provide multiple supports for long-range communication, and the intelligent processing of several WSNs information, even when they are not SIMPLE complaint.

This layer is achieved by the usage of a Machine-to-Machine platform that work on top of any IP or Bluetooth network, enabling the possibility of a proximity, long-distance or mobile networking scenario.

The bridging is based on data, since this layer is smart enough to gather it from the several WSN and generate events at its level. Also it makes possible for the information to be processed at the network level, enabling the generation of events in the offline state, without data being streamed from the WSN to the API.

3) SIMPLE Wireless Sensor Networks

Wireless sensor networks consist of large numbers of low-power sensing devices and at least one sink node that allow the connection from data subscribers.

There are several wireless sensor networks in a typical SIMPLE installation. The SIMPLE system has an innovative WSN platform that operates with subscriptions, having multi-hop geographical trusted routing and positioning capabilities. This platform implements some of the functions used in the SIMPLE self-organized middleware directly on the nodes and integrates them easily.

Other WSN platforms can be interfaced to the SIMPLE system by developing a suitable adapter. This adapter would be placed on a gateway device, presenting the WSN to the middleware through the same subscription and query mechanism as the SIMPLE variant.

B. Data Representation

Main processes of subscription starts from the application by setting up a subscription using the SIMPLE DSL. The DSL instructions are transformed into sensor network subscriptions using information about the network structure and associations with the data model. These sensor subscriptions are then distributed among the middleware nodes, which can take actions and distribute the query into the WSNs that they are attached to.

The subscriptions serve as a service agreement for the sensor nodes and how they may be acknowledged. The sensor nodes start providing data based on the requirements expressed by the subscription. Data travels to the middleware nodes, which then fuses and reformats it in accordance to the relevant request and information in the data model. Finally, the data is kept in data storage nodes inside the middleware to be, later or immediately, delivered to the client or clients.

1) XML Middleware Subscription

Sensor nodes, due to hardware and resource constraints, support only a limited and fixed set of XML tags. The XML messages follow a strict structure, where each can have only one parameter, called value. Value is a 32 bit signed integer.

Larger values can be transferred appending additional tags and appropriately structuring the messages. Content outside the value parameter is not supported. The number of tag-value pairs in a message is formally limited to 255, but practically WSN node radio messages will fit much less. Each XML tag needs to have a unique integer identifier associated with it in the converter database.

The first part of the subscription is about the issuer, the subscription id and the time that subscription is considered valid. When the receiver responds to a subscription the issuer and the subscription id values must be the same and its own identification must be added, in the provider value.

In the second part of the subscription, the requester sends information about the data to subscribe, separating it in spatial, semantic and temporal constrains. Spatial are related with the coordinates pair (X, Y) of the area to subscribe, semantic are related with sets of instructions that the provider must follow and temporal define the period timeouts for the provider to send responses.

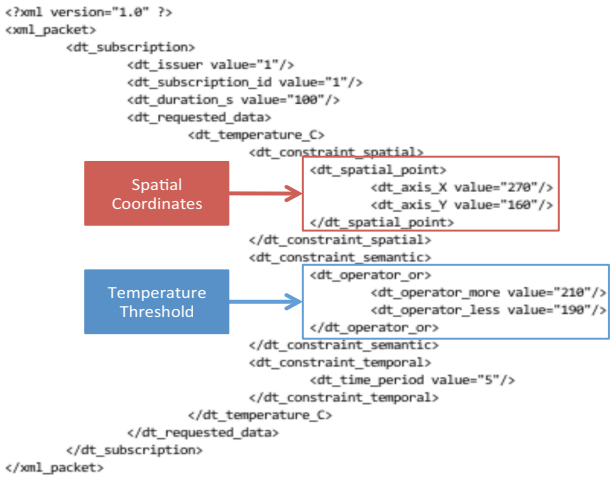

Figure 3. Subscription Example.

Following the example, to subscribe the temperature values in a room located at (270, 180), that are above 21 and below 19 Celsius degrees, in periods of five seconds, the requester will send a subscription like that of Figure 3.

Then, the provider returns a response with temperature value of 25 degrees Celsius that is a valid value defined by the requester, and although the provider isn't placed directly at the center of the coordinate pair, it can respond to the subscription since it is in its radius range. See Figure 4.

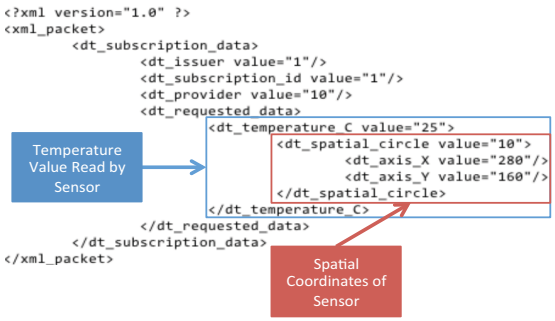

Figure 4. Subscription Response.

2) XML Binary Representation for WSN

A very compact data XML (binary) representation, which at the same time creates minimum overheads in terms of parsing, is used at the WSN level as provided by [9].

The binary representation of the XML operates on the principle of a table that relates all the tag names with a hex identifier and each tag can only have one optional integer parameter called value. The structure of the XML is preserved by using a triplet, which is comprehended of: (1) tag indexes, (2) subject fields, that reference the indexes, for storing the hierarchy and (3) value. Numeric values are encoded using the fewest possible bytes.

The middleware has methods to manipulate the binary messages, for converting from XML to XML binary and from xml binary to xml again. Also, there is available a C-language library supported by python scripts.

IV. SIMPLE APPLICATION PROGRAMMING INTERFACE

The SIMPLE API purpose is to hide the sensor network complexity and expose it with a generic API. To achieve that it has two main functions: (i) provide a generic API to all applications; and (ii) adapt business logic language to sensor level semantics:

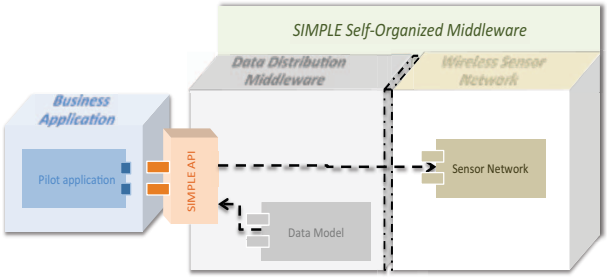

Figure 5. SIMPLE API Interface.

A. Generic Interface

Services published by the middleware can be used to manage, configure and query the underlying sensor network. These services can be exposed using the most suitable technology that the business application requires. By default, the SIMPLE API provides a SOAP web service implementation, but alternative solutions are possible too.

The most important services provided are query and subscription. Using the query interface it is possible to read current sensor data and stored sensor information or get sensor/device relationships. The subscription interface is used to get automatic responses triggered by events.

To notice that the query interface on the underlying network is a special subset of the subscriptions, it can be described as a one-time subscription.

B. Business Logic Adapter

The SIMPLE API is able of transforming business specific queries to language understood by sensor networks!

Classically, a sensor network has no knowledge of business processes, business items or business policies. So an important add-on provided by SIMPLE API is the translation of queries, written in a "business-friendly" language, to queries that can be executed by the sensor network, promoting a better usage of WSN on business domains.

The SIMPLE API uses the data model information, stored in the SIMPLE middleware, to map domain information, such as trucks, pallets and its constituents to the location of that is used by the Wireless Sensor Network, such as storing the coordinates of the cargo area of the truck, so its possible to translate subscriptions to the cargo area to actual WSN coordinates.

The SIMPLE API uses a domain specific language, the SIMPLE Query Language that can be used to express requests performed by the applications like "get the current temperature of box associated with shipping order N". Although this query is very close to the business needs it cannot be understood by the sensor devices, in fact, the sensor network implementation make use of a binary optimized XML-based language.

Using the domain model, which defines how the wireless sensor network maps to business terms, the SIMPLE API parses the business request and sends it to the wireless sensor network. For example, on a query like "warn if temperature in truck A goes over 30 degrees", the API goes to the domain model and matches "Truck A" with their wireless sensor network coordinates, and based on that, sends a subscription to the network. After receiving an event from that location it matches it again to a business term, such as the "Truck A".

The Application Programming Interface also provides services that can be used to configure the sensor network such as to registration of new sensors and their configuration.

V. DATA DISTRIBUTION MIDDLEWARE

The Data Distribution MiddleWare layer is used to connect the various applications to the wireless sensor networks. In order to accomplish that, it was created a Data Delivery Service Network (DDSN), depicted in Figure 6.

The DDSN is composed out of several Data Distribution MiddleWare (DDMW) elements and allows the integration between different Applications and Information FEEDs, providing long-range communication and heterogeneous information feed support.

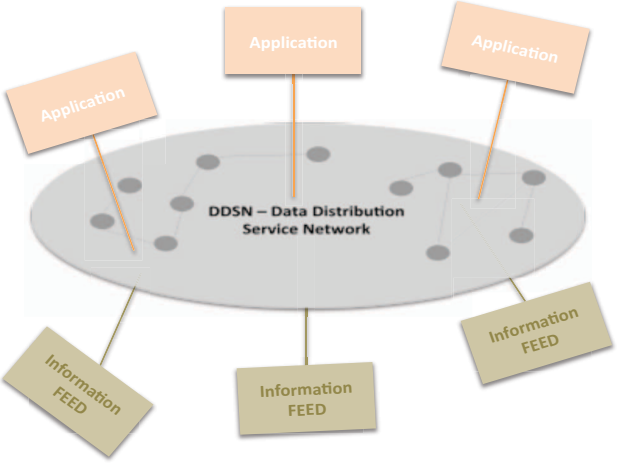

Figure 6. Data Distribution Service Network

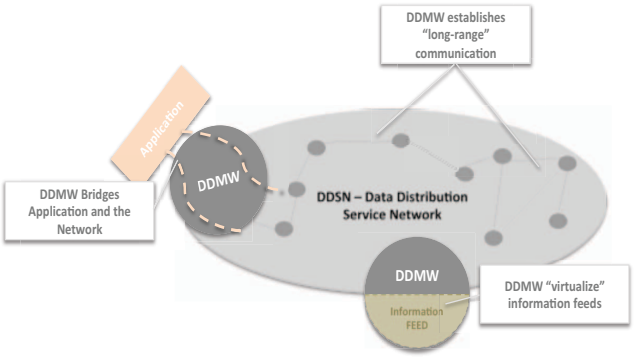

Figure 7. Data Distribution MiddleWare Purposes.

Figure 7. shows that DDMWs can have three purposes: (i) a bridge between the Applications and the network; (ii) a virtualization of the Information FEED, representing them within the network, and (iii) used as peers to allow that applications enquire Information FEEDs "far" from them.

Each DDMW is autonomous and operates independently and asynchronously of all other DDMW, is composed of two components: Network P2P Infrastructure and Data Services.

A. Network P2P Infrastructure

The DDMW Network P2P infrastructure is based on the JXTA [8] peer-to-peer protocol specification, and which implementation was initially developed in the scope of European research project FP7-216420 Cuteloop [7]. The infrastructure comprises three main components: Messaging Interface, Services' Indexer and Network Transport.

The Messaging Interface is simply used to send and receive messages; the Services' Indexer, acts in a distributed way to publish and discover the available services in the network; and the Network Transport handles all the possible communication technologies that can be used by the DDMW, choosing the right one considering the destination.

Network P2P Infrastructure grants zero configuration, which allows the automatic connection of a DDMW to a network, without manual operator intervention or special configuration servers, auto discovering other DDMW's to work together. Moreover, the DDMWs create an overlay transport network that allows the interaction amongst them even if some are behind firewalls, NATs, or use different network transports, such as TCP/IP or Bluetooth.

In addition, a Universally Unique Identifier (UUID) identifies each resource, allowing the location address of each DDMW to change since the UUID is kept, even if the node goes down and it comes back up again.

B. Data Services

Data Service component provides support to the subscriptions. Every subscription passed from the SIMPLE API to the DDSN is forwarded to the DDMWs that virtualize an Information Feed, where each one decides if its information feed is able to process that subscription. When an Information Feed detects the occurrence of a subscribed event, it sends the event through the DDSN back to the SIMPLE API to match the application that performed the corresponding subscription, as detailed in Figure 8.

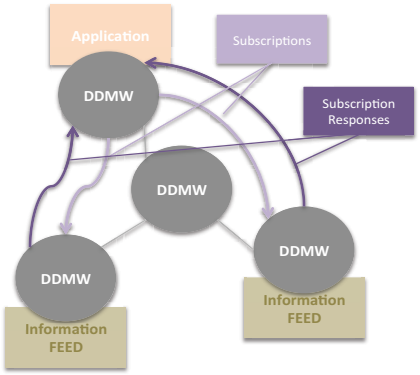

Figure 8. Data Distribution based on Subscriptions

In order to communicate with an application, every DDMW possesses a generic API that provides an interface that can be used by the SIMPLE API. This interface defines one method - sendSubscription(xmlSubs,callback) - that is used to send a subscription for specific data to an Information Feed and register an APICallback that is an interface with other method - responseRcvd(xmlResponse), used to process the data received in the subscription responses. This interface is implemented by the application that performs the subscription (SIMPLE API).

And to communicate with an Information Feed, a DDMW has a Connector that is specific to each type of Information Feed, implying a different implementation. A DDMW communicates with each Information Feed using the same behavior, to do so an abstract interface is available. The interface defines two methods: (i) sendResponse(xmlResp) that sends a response for the connected Information Feed; (ii) registerRcvr(type,rcvr) that adds a receiver for messages from Information Feed. InfoFeedRcvr corresponds to a callback with the method subsRcvd(xmlSubs), used to deliver the data sent by the Information Feed to DDMW interested in it. The xmlSubs represents the message to send.

An overview of the DDSN, where is represented the information flow induced by the introduction of a subscription and data from the information feeds, is detailed in Figure 9. In the top of the figure is an Application that connects to one DDMW in order to make a subscription. The DDMW will add the subscription to the database, which will be replicated through the DDSN. During this step, a callback to the application is locally associated with the subscription.

Each DDMW connected to an Information Feed will wait for the existence of a new subscription and verify if the information feed is able to process that subscription. If it is able, the subscription is sent to the Information Feed.

Every time a DDMW receives new data from an Information Feed, it is added to the database, and replicated. This new data may match the conditions of a subscription, so a creation of an event is needed in order to notify the application that made that subscription. This event can be created by any of the DDMWs present in the DDSN. The DDMW that creates the event will then add it to the database. Whenever the DDMW connected to the application detects a new event associated to one of its subscriptions, it sends that event using registered callbacks.

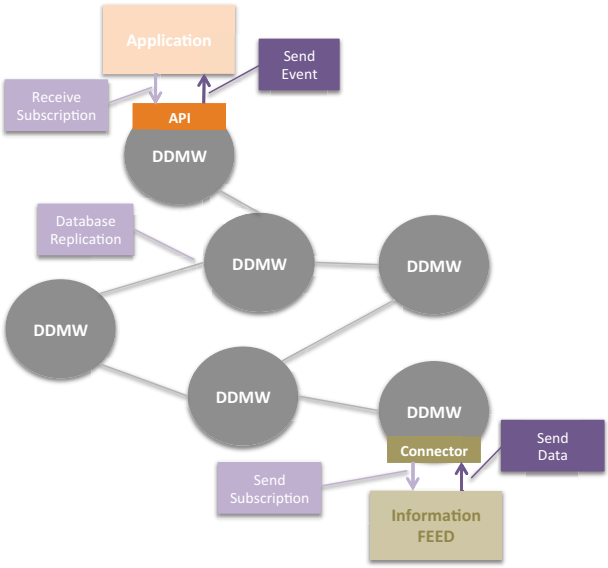

Figure 9. Information Flow inside the DDSN.

When a new subscription is made, it is interpreted by the DDMW in order to simplify it. This simplification consists in dividing, if necessary, the subscription based on the number of conditions. This simplification is performed in order to allow each DDMW connected to an Information Feed to verify if it can reply to that condition. This subscription partitioning is also performed in order to verify if any condition is already subscribed decreasing the number of subscriptions sent to the Information Feeds.

When a DDMW identifies that its Information Feed can gather information concerning the new subscription, it may require the translation of the subscription to the language used by it, in order to allow its computation. A similar translation may also be required by the DDMW to interpret the data received from the Information Feed. After that the DDMW identifies to which condition the received data refers to and inserts this information in the database.

Every time a DDMW detects new data insertion in the database, it tries to generate an event. To perform this process, the DDMW identifies all the subscriptions interested in that data, and: if a subscription is only interested in one condition and fulfills its requirements, a corresponding event is generated; otherwise, if the subscription is interested in several data, i.e. defines several conditions, then the remaining is searched in the database.

When a DDMW connected to an application finds an event that corresponds to a subscription associated to a callback in its local storage, that event is collected and delivered to the application using that callback.

VI. Wireless Sensor Network Layer

For individual nodes to adapt their behavior to the needs of an application, it is important that the node can interpret these needs, which will be in the form of subscriptions. As said, these subscriptions can ask for data from specific types of sensors, location, times, or with specific constraints on the value itself.

Thus, for the node to properly assess the relevance of a subscription to itself and adapt its behavior, it must at least have some notion of the sensors it is equipped with, some notion of its location, and some notion of time. Additionally, it must be able to evaluate semantic constraints on the values of data themselves. Finally, it must run some software that actually allows it to dynamically change its behavior. In SIMPLE implementation, this software is the MyriaChap.

A. *Implementation using MyriaChap*

MyriaChap [6] is a WSN platform for creating evolving and evolvable applications. The main functionality is that it allows individual nodes to change their behavior at runtime. This is achieved by breaking down functionality into small blocks (components), each implementing a basic function. Components can be created, deleted, and recombined at runtime by the node itself, thus allowing autonomous adaptation of nodes.

Compositions of components are constructed by letting components subscribe to each other, causing a subscribing component to receive input from another component. This way it will take the output of the other component as input for whatever its own operation is.

Components communicate and pass control among each other using messages that have some subject and content. This has two major advantages over a simple function calling mechanism. First, there is no need for one component to know the interface of other components, since it can always send any message, and it is up to the receiving component to decide what to do with it. Second, long and complicated chains of function calls won't emerge, which reduces the usage of RAM memory of the node.

To prevent the changing of component configurations during the lifetime of a node from causing memory fragmentation, MyriaChap also manages the use of memory by components, subscriptions and messages. It allocates a certain amount of memory for its own use at node start-up time, and reuses it whenever new structures are required.

VII. Conclusion and Future Work

Wireless Sensor Networks are the core element for the so-called sensing enterprises, by making available better and faster information that is relevant to the business. Now, this information may exist in a large set of sensors scattered around wide areas and that perhaps are not connected to a single WSN. So, it is needed to get information out of disparate heterogeneous WSNs comprehending distinct platforms (hardware types and/or software).

This paper purposes a way for making the link between these heterogeneous WSNs (comprising real-world objects) and business applications. The proposed solution is defined by a multi-layered architecture, composed of a well-defined API for interfacing with applications, an advanced solution for wireless sensor networks that provides processing using subscriptions, and a connection between them, able to conduct long-range distributed networked communications and data distribution and aggregations.

The solution enhances the reach and capacity of the IoT system by having many WSN participating in the system

without the need to changing the technologies already in place. The information exchange in disparate networks is achieved at the data level by translating the specific WSN data format into a unique standard throughout the system.

Moreover, in the absence of maestro nodes that have the capacity of continuously coordinate and organize the actions of other many nodes in a changing environment, not many viable alternatives exist in order of maintaining the network reliable. One of the other options nullifies the ad-hoc nature of a wireless technology, giving the control to a central node becoming dangerously vulnerable to changes in the environment. Thus, the self-organizing property and full-distributed nature of the solution provides robustness and enhanced service to Internet-of-Things systems/applications.

The middleware is being experimented and piloted in three, much different yet with much similar technological requirements, application cases: a smart-logistics application for military logistics tracking & tracing; a manufacturing-of-the-future application for monitoring and follow-up of white-goods production; and a smart-kitchen concept for localization and information of/about products.

A. Future Work

On-going research work focuses on providing semantic routing to messages and content in the network. Idea is the following: subscriptions passed on from Applications to the middleware network needs to be forwarded to the information feeds that is capable of processing it. When a MW receives a subscription, it forwards it to other MWs that virtualize information feeds, where then each one decides if its information feed is able to process that subscription. This implies the routing of subscription messages from MW to MW to reach-out for the MW that is virtualizing the WSN that can answer with the data. Routing of subscriptions based on semantics would greatly improve performance of message exchange in the MW overlay network. E.g. subscription like "notify me when comfort in room X goes is not ok" needs to be forwarded to WSN that is sensing room X (so concept of room and X needs to exist for routing) and also semantic notion of 'conform' needs to exist, even set as a function of temperature, humidity, noise, etc.

One other research path is dynamic support to Data Interoperability at the middleware level. The idea is to create middleware supports to enable plug-and-play approach to data interoperability in order for disparate data sources, which are represented in different formats and described in different languages, to just plug and interoperate within the environment. The middleware is to support the execution services for interoperability as well as the organization of interoperability related information across the whole systems (even if in distributed fashion).

For the viewpoint of adherence to standards the plan is to develop supports for oneM2M/ETSI M2M standards. These standards define a standardized horizontal Machine2Machine service capability layer on top of the connectivity layers that does registration, access rights, security & authentication, data-transfer (containers), subscribe/notify, groups, etc. This will enable interoperation with more middleware, sensors and systems within technology-richer IoT environments.

ACKNOWLEDGMENTS

This work was supported in part by national funds, notably by the Portuguese FCT - Fundação para a Ciência e a Tecnologia, and co-funded by the European ARTEMIS Joint Undertaking for Embedded Systems, under the scope of the FCT/ARTEMIS-100261 SIMPLE research project. Work was also supported by European Commission funds under the framework of FP7-288315 PROBE-IT Support Action.

REFERENCES

[1] R. Stevens, K. Kalaboukas and M. Forcolin, "Simple Sensor Network Middleware and FADs", in Proceedings of the 17th International Conference on Concurrent Enterprising (ICE 2011), 2011, pp. 1-9.

[2] K. Georgouleas, K. Kalaboukas, A. Población‐Hernández, M.García‐Otero and P. Malo, "A Hierarchical Network Architecture for Intelligent Manufacturing and Logistics", in Proceedings of the Embedded World Conference 2012 (EWC2012), 2012, pp. 1-11.

[3] D. Miorandi, S. Sicari, F. De Pellegrini and I. Chlamtac, "Internet of things: Vision, applications and research challenges", Ad Hoc Networks, Vol. 10, Issue 7, pp. 1497-1516, Sep. 2012.

[4] I. F. Akyildiz and I. H. Kasimoglu, "Wireless sensor and actor networks: research challenges", Ad Hoc Networks, Vol. 2, Issue 4, pp. 351-367, Oct. 2004.

[5] F. Ye, H. Luo, J. Cheng, S. Lu and Lixia Zhang, "A two-tier data dissemination model for large-scale wireless sensor networks", in Proceedings of the 8th annual international conference on Mobile computing and networking (MobiCom'02), 2002, pp. 148-159.

[6] F. van der Wateren, "The art of developing WSN applications with MyriaNed". Chess Company, the Netherlands, Tech. report, 2008.

[7] H. Sundmaeker and T. Kovacikova, "CuteLoop - An Approach for Networked Devices Enabled Intelligence", in Proceedings of the Fifth International Conference on Software Engineering Advances (ICSEA), 2010, ISBN: 978-1-4244-7788-3, pp. 205-212.

[8] Li Gong, "JXTA: a network programming environment", in IEEE Internet Computing, Vol.5, Issue 3, pp.88-95, May/Jun 2001.

[9] J. Preden, and R. Pahtma, "Exchanging situational information in embedded networks", in Proceedings of 2nd International Conference on Adaptive Science & Technology (ICAST 2009), 2009.

[10] Y. Zheng and J. Cao, "Highly Scalable and Efficient Publish/Subscribe Protocols Using Geographic Information for Wireless Sensor Networks", in Proceedings of the 3rd International Conference on Intelligent Sensors (ISSNIP 2007), 2007.

[11] K. Aberer, M. Hauswirth and A. Salehi, "Infrastructure for Data Processing in Large-Scale Interconnected Sensor Networks", in Proceedings of 8th International Conference on Mobile Data Management (MDM'07), 2007, ISBN: 1-4244-1241-2, pp. 198-205.