Efficient Resource Management based on Non-Functional Requirements for Sensor/Actuator Networks

C.Timm[1], F.Weichert[2], C.Prasse[3], H.Müller[2], M.ten Hompel[4] and P.Marwedel[1]

[1]Department of Computer Science 12, TU Dortmund, Germany
[2]Department of Computer Science 7, TU Dortmund,Germany
[3]Fraunhofer Institute for Material Flow and Logistics, Dortmund, Germany
[4]Chair for Materials Handling and Warehousing, TU Dortmund, Germany
e-mail: constantin.timm@postamt.cs.tu-dortmund.de

Abstract

In this paper, a novel resource management approach is presented for publish-subscribe middleware for sensor/actuator networks. The resource management was designed with the possibility to add non-functional requirements at runtime to subscription messages. This approach allows utilizing service level agreements that can then be employed in order to guarantee a certain quality of service or to reduce the energy consumption of a sensor node in a sensor/actuator network. As an example, a sensor/actuator network for facility logistics system (a conveyor belt system) is evaluated with respect to energy consumption. This sensor/actuator network is mostly controlled by image processing based sensor nodes. It is shown that an adaptive processing interval for these sensor nodes can reduce the energy consumption of the entire network. The utilization of non-functional requirements allows the system to adapt -- after software development -- to context changes such as the extension of the conveyor belt systems topology.

Keywords

Middleware, Non-Functional Requirements, Sensor/Actuator Network

1. Introduction

Web service based technologies are becoming a standard technique for connecting embedded systems. Especially the spreading of DPWS (Device Profile for Web Services) (Chan et al., 2006) in this field and its utilization of standard internet protocols shows that standardization is a major promoter of scalable and re-usable SANETs (Sensor/Actuator NETwork). SANETs provide the possibility to gather information from an environmental context via sensors and sensor nodes and to interact with the environment through actuator nodes without a central control infrastructure – often in a wireless environment (Akyildiz and Kasimoglu, 2004) but not restricted to this. The benefit of these networks is a high adaptation capability in terms of deployment and of failure recovery.

One of the major features of DPWS is the specification of a web service based publish/subscribe paradigm which is known from state-of-the-art automotive communication protocols or from factory steering components such as PLCs (Programmable Logic Controllers). The basic principle of this publish/subscribe

paradigm is that there are service providers which push information about an event only to those service consumers that subscribe to the corresponding service providers in advance. Especially when processing capability and network bandwidth are scarce, publish/subscribe has an advantage in comparison to a polling-based communication because the transmission of data is only initiated when necessary. A state-of-the-art middleware which is based on DPWS is MORE (network-centric Middleware for GrOup communication and Resource Sharing across Heterogeneous Embedded Systems) (Wolff et al., 2007).

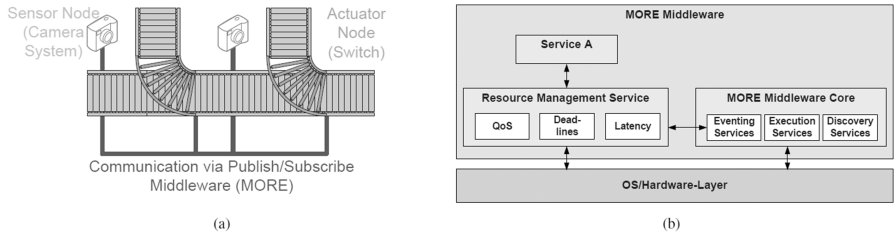

Figure 1: (a) SANET-based Conveyor Belt System and (b) Integration of Resource Management with Subscription Manager

According to (Pavlovski and Zou, 2008), ”*non-functional requirements are ... referred to as constraints, softgoals, and the quality attributes of a system”*. This non-functional ”information” should be modeled in a middleware architecture to handle resource utilization efficiently. As stated in (Franch and Botella, 1998), software design comprises three non-functional core concepts. First of all, there are non-functional attributes which comprise certain attributes such as *time efficiency*. The second non-functional concept is a non-functional behavior which is the assignment of a non-functional attribute to a software component. The last non-functional core concept, is that of a non-functional requirement which is the actual assignment of a concrete value to a non-functional behavior. In many cases a non-functional requirements models a QoS (Quality of Service) requirement. The newly designed central point for gathering non-functional requirements and for controlling the adaptation to these requirements is the extended resource management, named NOFURES (NOn-FUnctional RESource management). In contrast to the original resource management service of MORE, NOFURES allows to assign non-functional attributes and requirements to the subscription mechanism of MORE. As opposed to other software design methods, the non-functional requirements from the actuator nodes are evaluated on each sensor node in a middleware environment at runtime.

In this paper, an exemplary SANET for a conveyor belt system from the field of automated facility logistics systems is the considered use case (Figure (a)). The SANET of the conveyor belt system comprises different sensor nodes and actuator nodes (Timm et al., 2011). The sensor nodes of the system (e.g. camera system or RFID readers) are directly connected to the actuator nodes such as a deflecting belt or switch employing publish-subscribe methods of MORE, as was presented in (Timm et al., 2011). This is fundamentally different – but more efficient – compared to traditional systems where there is only a central controlling instance. The functionality of NOFURES is employed and evaluated with respect to this exemplary

SANET. Constraints and non-functional attributes of the SANET controlling the conveyor belt systems can be, for instance: soft deadlines, analysis quality and input quality.

The most relevant resources at processing level are execution time and energy consumption. The first was already addressed in former versions of MORE (Alonso, 2010) while the latter is a new objective which is considered in this paper. One of the most critical parameters in terms of energy consumption of a service provider is the *update interval* in which events have to be processed and how often a service consumer needs that information. This *update interval* is directly connected to the QoS of a middleware service and therefore, this non-functional attribute is considered in this paper. The specification of a non-functional requirement allows to adapt the system behavior at runtime.

The major contributions of this paper can be summarized as follows:

- Non-functional requirements are taken into account in a publish/subscribe middleware at runtime.
- The specification of a central resource management enables controlling/observing the QoS of all middleware services.
- The energy consumption of sensor nodes is explicitly considered at runtime.

The paper is structured as follows: After this introduction, related work is presented in Section 2. The principles of embedding non-functional requirements to the subscription mechanism of a DPWS-based middleware are introduced in Section 3. In Section 4, the results are presented, followed by a conclusion in Section 5.

2. Related Work

In view of the enormous number of publications in the domain of SANETs as well as in the field of the SANET specification, the following presentation focuses on papers that are related to the approach presented in this paper. In (Sharaf et al., 2004), the authors presented an approach to reduce the amount of data which is transferred in a wireless sensor network over the course of time. The authors – in contrast to this paper – mainly focused on the routing in a network and on the aggregation of data inside the network to achieve savings in terms of energy. The authors of (Munir and Filali, 2007) described a routing and topology building methods for a wireless SANET. The proposed method models the end-to-end delay and the energy consumption as hard constraints which must not be violated. In addition to that, the topology of the network is constrained such that there is only one connection to an actuator node.

The methods proposed in this paper adapt the processing frequency of the sensor nodes in the network which can be seen as a type of Adaptive Sampling (Alippi et al., 2007). Within that work the authors proposed an adaptive sampling method that enabled the developer of a wireless sensor network to save energy. The approach took into account that some wireless SANETs, the energy consumption for processing is higher than for communication. This fact is also exploited by the

methods in this paper. However, the work presented in (Alippi et al., 2007) does not focus on the same application environment which are Ethernet-based SANETs.

Another area of interest is the field of resource management (Alonso, 2010) and adaptive applications (Davies et al., 1996). In the latter work, the authors presented a framework for creating a sensor network with different QoS levels.

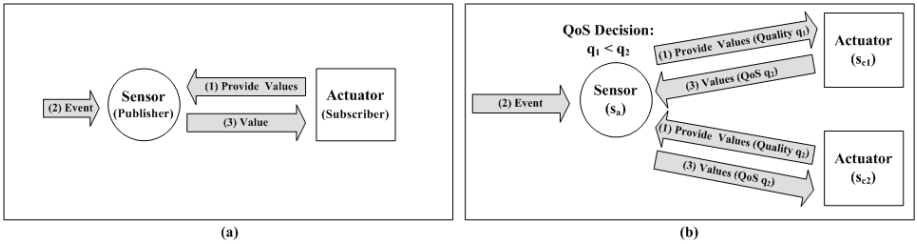

Figure 2: Publish-Subscribe-Mechanism without (a) and with (b) the ability to specify QoS requirements

The last areas of interest are business applications and software architecture. Several papers about non-functional requirements exist in these areas (D'Ambrogio, 2005; Pavlovski and Zou, 2008; Franch and Botella, 1998) but they are more software architecture related and do not focus on runtime adaptivity. Overall, it can be summarized that not all aspects of resource management in the area of SANET research are exploited.

3. Non-Functional Requirements Aware Middleware

The capability to adapt sensor node behavior in a system to non-functional requirements and its integration in a resource management (NOFURES) are the novelties presented in this paper (Section 3.2). Section 3.1 will summarize methods of the MORE middleware (cf. Figure (b)) which will be followed by the description of a exemplary implementation for a conveyor belt system in Section 3.3.

3.1. General Architecture

The design of the MORE Middleware (Schmutzler et al., 2008; Wolff et al., 2007) conforms to the paradigm of SOA (Service Oriented Architectures) by adopting it for external and internal communication to provide a high degree of flexibility. In terms of communication protocols, MORE satisfies a subset of the DPWS specification (Chan et al., 2006) including an ad-hoc service publication and service discovery mechanism which is specified in the standard WS-Discovery. The latter is a multicast-based approach which is well-suited for local area networks. For the use in wider network topologies, a concept called discovery proxy is described which hosts a directory service. The major contribution of the MORE middleware are the Added Value Services which offer common functionality like service orchestration, group and resource management services (refer to (Wolff et al., 2007) for more details). The two most important features of this middleware are the resource management – an extension of it is presented in the next section – and the publish/subscribe

eventing services as specified in the DPWS specification (Chan et al., 2006). The latter extends web services by the possibility to subscribe to asynchronous event messages of a service and is based on the WS-Eventing standard. The use of the publish/subscribe paradigm (cf. Figure 2(a)) within the MORE Middleware is especially important for sensor/actuators networks and makes the design and deployment of such networks smarter. A polling-based approach is inappropriate in SANETs where nodes with low processing capabilities or with energy constraints can be found.

```
<SOAP-ENV:Envelope ... xmlns:nonfunc="...">
  <SOAP-ENV:Header>
    ...
    <nonfunc:Parameters>
      <nonfunc:item>
        <nonfunc:Name>notify_int_low</nonfunc:Name>
        <nonfunc:Value>490</nonfunc:Value>
        <nonfunc:Unit>ms</nonfunc:Unit>
      </nonfunc:item>
      <nonfunc:item>
        <nonfunc:Name>notify_int_high</nonfunc:Name>
        <nonfunc:Value>510</nonfunc:Value>
        <nonfunc:Unit>ms</nonfunc:Unit>
      </nonfunc:item>
    </nonfunc:Parameters>
  </SOAP-ENV:Header>
  ...
</SOAP-ENV:Envelope>
```

Figure 3: Integration of Non-Functional Requirements to Subscription Message

3.2. Non-Functional Resource Management – NOFURES

The novel resource management service has the ability to cooperate with the subscription management of MORE. The new functionality is summarized by the term NOn-FUnctional RESource management (NOFURES). A tight integration with the middleware core and the operating system enables the resource management service (cf. Figure (b)) to handle the non-functional requirements added to the subscription of a service. The specification of these requirements of the subscriber enables the resource management to adapt the behavior of the system towards a SLA (Service Level Agreement). NOFURES is designed to track several resources at runtime. The non-functional attributes are specified using the WSDL service description; analogue to the specification of performance qualifier for web services proposed in (D'Ambrogio, 2005). An example subscription message with non-functional requirements to the update interval of the service consumer is depicted in Figure 3. As one can see, non-functional information is added to the header of the subscription message. The integration was accomplished in a way that these messages can also be interpreted by DPWS devices which do not need non-functional requirements. A new namespace `nonfunc` was created in which the non-functional requirements can be specified. The non-functional requirements are listed in an XML sequence called `nonfunc:Parameters` which includes one or more non-functional items comprising a `name`, a `value` and a `unit`. The requirements listed in Figure 3 show lower (490ms) and upper bounds (510ms) for the update interval. The adaptation of the behavior of the service to SLA is controlled by NOFURES. In particular, the resource management service handles all accesses of a service to the underlying operation system and libraries and controls the execution of the service.

The control functionality can include features such as the (average and worst case) runtime of a service, the quality of the result of a service or service failures. This is needed in order to regulate the processing of a service towards a conformance to the specified non-functional requirements. The traditional publish/subscribe process is depicted in Figure 2(a). The actuator nodes subscribe to the sensor node events and get informed when new events of the subscribed type happen. In comparison to that, an example of how an SLA or QoS Decision can be used with NOFURES is depicted in Figure 2(b). For example, there could be service consumers s_{c1} and s_{c2} which are interested in a certain service s_a providing an image analysis with certain frame rates $x_1 < x_2$ with respective QoS of $q_1 < q_2$. With NOFURES s_{c1} and s_{c2} can now inform s_a with which particular QoS the results of s_a are required (e.g. s_{c1} needs q_1 and s_{c2} needs q_2). For the QoS, several policies could be applied, e.g. provide a service with a QoS that satisfies all requirements. If the latter is applied, the NOFURES service on s_a can choose q_2 for both services in order to satisfy the non-functional requirements of s_{c1} and s_{c2}.

3.3. Use Case: Camera-based Conveyor Belt System

As an exemplary system, a SANET controlling a conveyor belt is considered. The most important places within a conveyor belt system are the switches taking the decision to route a parcel to one or another direction. In the past, these switches were controlled by a larger number of sensors, such as light-barriers or RFID-readers and a central control instance that tracks all parcels on the conveyor belt and which is responsible for taking control decisions. This approach is inefficient, especially from the perspective of costs but also from the point of flexibility and scalability. A new approach was introduced in (Timm et al., 2011) which replaces light-barriers and RFID-readers at the switches by low-cost cameras and an in-situ marker detection system (as depicted in Figure (a)). The employed marker technology is called QR code (International Organization for Standardization, 2006). The image processing as part of the marker detection system was accelerated by a parallel processing hardware based on OpenCL (Khronos Group, 2010). The topology of the SANET is as follows (cf. Figure (a)): The sensor nodes observe one or more switches and the belt in front of them. The actuator nodes of the SANET are the switches which subscribe to the sensor nodes. All sensor nodes and actuator nodes are equipped with the MORE middleware and the new resource management NOFURES.

One of the most critical parameters in terms of energy consumption of a service provider is the interval in which events take place/have to be processed and how often a service consumer needs that information. If, for instance, the information (QR code) which is provided to the service consumer is not updated in consequent events, this information need not be transferred again. In terms of the conveyor belt, this could be a parcel which is still on the same trail towards a switch. The switch as a service consumer is only interested in the parcel's data if it represents new information and therefore attributes with the necessary update interval are added to the sensor node's subscription. The sensor node can then adapt to this requirement.

Therefore, the non-functional attribute which occurs on the sensor nodes and which is evaluated in this study, is the update interval. It describes the minimal and maximal time interval in which a sensor node has to provide data if events occur. NOFURES can actively restrain the events which are published by a sensor node. If more than one actuator node subscribes to a sensor node. The minimal update interval is chosen.

4. Evaluation

This section provides the basic requirements to evaluate the functionality and efficiency of the proposed resource management in real logistics system architecture and the corresponding results. First of all, the testbed for the sensor nodes is introduced (Section 4.1) and then the results are presented (Section 4.2). The evaluation shows how resources such as energy can be saved by providing non-functional requirements to the subscription process.

4.1. Testbed

Both, energy consumption and performance are measured with a performance and energy benchmarking testbed. The sensor node is powered via a 5V DV power supply connection. For measuring the power consumption of the sensor node, a power clamp at the 5V power line is utilized. The power clamp provides a voltage proportional to the current flowing through the probed lines which can be measured employing an oscilloscope (Sampling frequency: 10kHz).

The following tests were conducted: The energy consumption and the processing time were measured for two image sizes: 320×240 pixels and 640×480 pixels. After these initial tests, several update intervals were tested in order to determine the update interval with optimal energy consumption for the considered conveyor belt system. The update intervals of the sensor nodes are 250ms±10ms, 500ms±10ms and 1000ms±10ms. They are added as lower and upper bounds to the subscription messages as depicted in Figure 3. The maximal speed of the considered conveyor belt system is approximately 1 meter per second. Due to this and the architecture of the system, the largest possible update interval is 1000ms±10ms, otherwise not all parcels can be identified properly. This is the standard speed of conveyor belt systems.

The baseline system configuration is publish/subscribe SANET without any restriction in terms of detected QR codes. Every detection is therefore transmitted.

4.2. Results

The results in Figure 4 show the different processing phases of the sensor nodes and thereby prove that theses nodes can adapt to the desired update intervals. While waiting for the next image, the power consumption ranges from 3.5 up to 4.5 watts. During image processing, up to 6.75 watts are consumed by the sensor node. In this figure, only processing intervals are shown where a QR code was fully decoded. Therefore, the execution times for images with incomplete QR code detection are shorter. On the other hand, image processing methods and most of the detection-

related algorithms are executed, regardless of the presence of a QR code in the image.

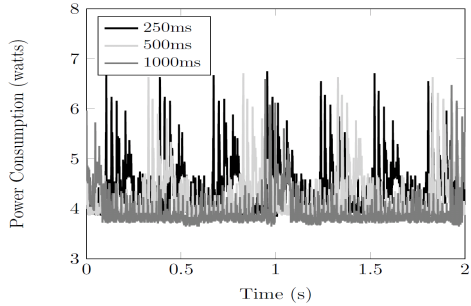

Figure 4: Power Consumption for Different Subscription Intervals

Table shows the results for energy consumption and the processing time for different image sizes. When an QR code is detected in a image, analysis required 88ms for a 320×240 pixels image and 291ms for 640×480 pixels image (energy consumption: 0.377 Joule, respectively 1.271 Joule). The lower bound for image processing (no QR code in an image) within the marker detection process for a 320×240 pixels image amounts to 36ms and to 132ms for a 640×480 pixels image. The minimal energy consumption for these lower processing bounds amounts to 0.152 Joule, respectively 0.574 Joule.

Image Size (Pixels)	QR code Detection		No QR code detected	
	Avg. Runtime (s)	Avg. Energy Consum. (J)	Avg. Runtime (s)	Avg. Energy Consum. (J)
320×240	0.088	0.377	0.036	0.152
640×480	0.291	1.271	0.132	0.574

Table 1:Energy Consumption for Analyzing a Single Image

The energy consumption results for the sensors nodes and the different update intervals are depicted in Table 2. The energy consumption values comprise a one minute time frame. Minimal energy is consumed when no QR code was processed in the specified interval and maximal energy is consumed in the specified interval when for each processed image a QR code was recognized. When no update interval was specified in a subscription message then minimal energy consumption for an image of size 320×640 pixels is 196 Joule and 211 Joule for an image of size 640×480 pixels. The maximal energy consumption for the same scenario is 212 Joule (320×240) respectively 225 Joule (640×480). These energy consumption values also characterize the system without NOFURES and therefore, they are used as the baseline for the evaluation. For an image size of 640x480 pixels and an update interval of 250±10ms, an evaluation not possible since the average runtime of QR detection exceeds this interval with 290ms. For the largest update interval (1000±10ms), the minimal energy consumption for an image of size 320×240 pixels is 183 Joule and 187 Joule for an image of size 640×480 pixels. The maximal energy

consumption for the same scenario (1000±10ms) is 187 Joule (320×240) respectively 200 Joule (640×480). This means that, in total, 12% (187Joule/212Joule) energy is saved in a system working with images of size 320x240 pixels and an update interval of 1000ms±10ms, in comparison to a system without NOFURES. For 640×480 pixels sized images, there is a reduction of the energy consumption of up to 8% (200Joule/225Joule). Overall, it can be summarized that by extending the update interval of the sensor nodes energy can be saved.

Image Size (Pixels)	Interval (± 10 ms)	Minimal Energy Consumption (J)	Maximal Energy Consumption (J)
320×240	Without NOFURES	196	212
320×240	250	188	202
320×240	500	185	191
320×240	1000	183	187
640×480	Without NOFURES	211	225
640×480	250	-	-
640×480	500	194	218
640×480	1000	187	200

Table 2: Energy Consumption per Minute for Different Update Intervals

5. Conclusion

This paper described an enhancement of a publish/subscribe middleware by the utilization of non-functional requirements for enabling a more efficient utilization of resources such as energy in a sensor/actuator network. To this end, the resource management of a state-of-the-art middleware based on DPWS, called MORE, was expanded to control and track the behavior of running services. Furthermore, it was extended to evaluate non-functional requirements encapsulated in subscription messages. The subscription process of the MORE was modified to specify non-functional requirements by a service consumer and to interpret them on the service provider side. NOFURES was then applied on an automated facility logistics system. The SANET in this system was employed with MORE and NOFURES. It was evaluated towards it capability to save energy by specifying a certain event notification interval. For future work, several other QoS/non-functional requirements will be evaluated with NOFURES. Furthermore, it will be evaluated if it is even more beneficial for energy consumption of sensor nodes when they share information, e.g. a preceding sensor node can share information whether there are parcels on their way to succeeding sensor nodes.

6. References

Akyildiz, I. F. and Kasimoglu, I. H. (2004), “Wireless sensor and actor networks: research challenges”, *Ad Hoc Networks Journal (Elsevier)*, Vol. 2, No. 4, pages 351 – 367.

Alippi, C., Anastasi, G., Galperti, C., Mancini, F., and Roveri, M. (2007), “Adaptive sampling for energy conservation in wireless sensor networks for snow monitoring applications”, *IEEE International Conference on Mobile Adhoc and Sensor Systems*, pages 1 –6.

Alonso, A., Salazar, E., and Lo andpez, J. (2010), "Resource management for enhancing predictability in systems with limited processing capabilities", *IEEE Conference on Emerging Technologies and Factory Automation*, pages 1 –7.

Chan, S. et al. (2006), Device Profile for Web Services, OASIS Standard.

D'Ambrogio, A. (2005). "A WSDL extension for performance-enabled description of web services". *International Conference on Computer and Information Sciences*, pages 371–381.

Davies, N., Friday, A., Blair, G. S., and Cheverst, K. (1996), "Distributed systems support for adaptive mobile applications", *Mobile Networks and Applications*, Vol. 1, No. 4, pages 399–408.

Franch, X. and Botella, P. (1998), "Putting non-functional requirements into software architecture", *International Workshop on Software Specification and Design*, pages 60 –67.

International Organization for Standardization (2006), "Norm ISO/IEC 18004 2006. QR Code 2005 bar code symbology specification"

Khronos Group (2010), "OpenCL Specification"

Munir, M. F. and Filali, F. (2007), "Maximizing network-lifetime in large scale heterogeneous wireless sensor-actuator networks: a near-optimal solution", *Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*, pages 62–69

Pavlovski, C. J. and Zou, J. (2008), "Non-functional requirements in business process modeling", *Asia-Pacific Conference on Conceptual Modelling*, pages 103–112.

Sharaf, A., Beaver, J., Labrinidis, A., and Chrysanthis, K. (2004), "Balancing energy efficiency and quality of aggregate data in sensor networks", *VLDB Journal*, pages 384–403.

Timm, C., Weichert, F., Fiedler, D., Prasse, C., Müller, H., ten Hompel, M., and Marwedel, P. (2011), "Decentralized control of a material flow system enabled by an embedded computer vision system", *IEEE ICC RWFI*, pages 1–5.

Wolff, A., Michaelis, S., Schmutzler, J., and Wietfeld, C. (2007), "Network-centric middleware for service oriented architectures across heterogeneous embedded systems", IEEE *EDOC Conference Workshop*, pages 105–108.