Sharing sensor networks

Manabu Isomura[1], Till Riedel[2], Christian Decker[2], Michael Beigl[2,3], Hiroki Horiuchi[1]
[1]*KDDI R&D Laboratories* [2]*TecO / University of Karlsruhe* [3]*TU Braunschweig*
{isomura, hr-horiuchi}@kddilabs.jp *{riedel, cdecker, michael}@teco.edu* *beigl@ibr.cs.tu-bs.de*

Abstract

Many industrial applications rely on sensors and sensor networks residing on machinery, transport containers or in the environment. For distributed processes in such domains the sharing of those sensor networks is crucial. This paper introduces a peer-to-peer (P2P) architecture for sharing services provided by existing sensor networks with any Internet based application. The differences between the sensor networks are abstracted using a service-oriented approach. The proposed technology is able to build up a global sensor Internet across multiple domain boundaries. Our implementation is evaluated with 300+ sensor nodes organized in a P2P network across continents.

1. Introduction

Recently, we observe the deployment of many wireless sensor networks (WSN). Successful trials were conducted in environmental, habitat and structural monitoring, industrial scenarios and home appliances [2]. As the number of platforms increases, we realize the splitting in more specialized platforms than general purpose ones. However, truly ubiquitous applications will exploit the capabilities of several diverse sensor networks. As a motivating example for such applications, we consider distributed manufacturing. Hereby, locally distributed enterprises form alliances in order to combine their resources and collaboratively operate as a larger, but more flexible manufacturing entity. As a result, flexible manufacturing processes lead to an increased product variety and allow price-worthy customized products [1].

This paper provides the following contributions enabling the use of WSNs in these processes. We propose and implement a uniform way to access and present different services from various sensor networks by utilizing the Universal Plug and Play (UPnP) [6] standard. Furthermore, we propose and implement an approach to combine these services in an application-specific and network-independent manner. This approach bridges peer-to-peer (P2P) networks and wireless sensor networks.

In the following section we analyze the application specific and general technical requirements for the system. Then we provide a proposal for our P2P and UPnP based architecture in section 3. In section 4 we describe a prototype implementation of the system where arbitrary services can be specified and instantiated on a remote sensor network respectively. Section 5 evaluates the implementation based on an application trial and we finally conclude at section 6.

2. Analysis

Applications of distributed manufacturing are supported by distributed manufacturing systems (DMS), which allow the sharing of processes and present resources and information from partner enterprises as "own" resources. DMS were proposed, simulated and implemented during last years [3]. But, to our knowledge none of them utilizes wireless sensor networks. However, especially in distributed manufacturing processes we expect a growing need for information from multiple sources of specialized sensor networks. In the following we explore how sensor networks can contribute to a DMS and what technical problems still have to be solved to enable an effective use of WSN technology in this application area.

2.1. Application Analysis

Sharing sensor networks yields several advantages for DMS. Firstly, sensor networks allow a continuous and direct monitoring of items during the manufacturing process. For instance, an electronic sensor seal can be implemented which constantly checks for potential violations of storage and delivery conditions while items are in transit [5]. Secondly, through specialization in manufacturing more and more parties are involved, e.g. producers, freight carriers, suppliers etc. As a consequence, detailed and electronically processable information originating from various sites is required to ensure in-time delivery and production quality throughout the process. Sensor networks may automate processes by seamlessly share detailed process information at the point where it is really needed. Thirdly, monitoring and automatic processing of online data from sensor data

across large sites allows manual maintenance inspections to be reduced to a minimum, leaving the manufacturing process as automated as possible. Manual action is then only required in cases of unrecoverable exceptions.

Figure 1 illustrates an example process sharing information from different wireless sensor networks. Each part of the manufacturing process resides in a different location and utilizes different types of sensor nodes. In Factory A, the manufacturing process is sensitive to temperature. A deployed sensor network of temperature sensors around the items monitors, that certain ranges are met. After the transport of the items to Factory B, the process now becomes sensitive to humidity. Humidity sensors deployed in Factory B check for violations of given thresholds. Throughout the process, quality management is always directly applied on the items during the manufacturing. Sharing this information contributes to the guarantee that the product complies to required quality at the end of the process chain. The benefit and cost of sharing information must be calculated in an economic way by comparing the increased quality and maintenance efficiency to the technology costs related to the revelation of process details.

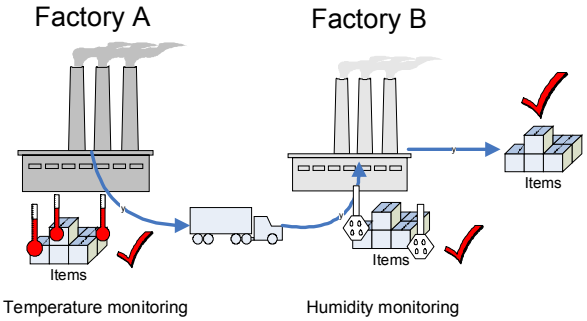

Figure 1. Example of a distributed manufacturing process

2.2. Technical Analysis

In this short analysis, we like to explore some important prerequisites necessary for the sharing of resources and services among sensor networks. The identification of these prerequisites is guided by the analysis of the above implementation scenario. In summary, from this analysis we identified three problem areas that we see as most important and which we address in the solution presented in this paper:

Connectivity. Sensor systems deployed on different manufacturing sites often utilize non-compatible, proprietary low-level communication protocols [2]. Trivially, *sharing of information presupposes common network connectivity* so that sensor and actuator networks on all sites provide a gateway to one common networking infrastructure.

Heterogeneity. This problem arises from the diversity of information presentation in factories and/or organizations involved in a DMS. E.g. sensor network platforms may use different data format and no common description of the information has been established yet. Applications, like monitoring applications, need a clear interpretation of data from multiple WSNs.

Consequentially, *heterogeneity demands a standardized abstraction* to enforce understanding between the different expressions of sensor information.

Interoperability. The third problem is how to flexibly bridge and combine information streams from remote sensor networks. Flexible manufacturing processes require that sensors networks in Factory A and B can be ad-hoc combined with other sensor networks and systems in order to support various processes. Our conclusion is that *flexible bridging and combining needs an overlay network* to enable such combination of remote processes.

Confidentiality. When designing systems that communicate data across multiple companies, and thus security domains, controlling the information flow is an obvious problem. Therefore, we finally see the need for *data centric access control*, that ensures that only data critical to the distributed process is forwarded.

2.2.1 Connectivity

Sensor networks fulfill very specific requirements concerning low-power communication and node interconnectivity. For this purpose many RF interfaces and optimized low-level access channel protocols access exist in parallel. E.g. the Particle platform [4] is available for a number of PHYs such ZigBee, Bluetooth plus some proprietary protocols using the 868 and 315 MHz band.

To enable full connectivity a client has to be introduced to the sensor network. For one type of sensor network we can add the RF interface to the client enabling it to connect to the sensor network on the PHY layer. However, if we consider accessing a wide variety of sensor network platforms, the client has to support each single RF interface. As a solution to this problem WSN traffic is bridged on a low-level to IP networks. By exposing the complete WSN directly to LAN or even WAN consequent problems may arise, e.g. that security and privacy must be mapped and enforced in an IP network. This makes it necessary to restrict low-level access to a reasonably small host network. This, however, stays in contrast to our goal of full connectivity. Therefore, we propose the use of high-level protocols that can retain the semantics of sensor network communication by encapsulating in widely supported communication standards.

2.2.2 Heterogeneity

Accomplishing connectivity to a wide range of WSNs reveals their true heterogeneity. The client now has to interface a number of different platforms hosting different sensor types and using different data encodings.

Data packets in sensor networks are often built in a way to support cross-layer protocol optimizations. For this purpose the data is encapsulated in an efficient encoding that can easily be parsed by the system. Because of different protocol stacks and operating systems, this leads to very different presentations of structured information in a packet. As an example the Particle platform use a tuple oriented data format enforcing strictly typed information. Motes use Active Messages allowing a direct mapping to the component interfaces of TinyOS. A client would have to support all different encodings in such a heterogeneous environment.

However, even if we can understand the message encoding, we will still fail to extract sensible information from the data. If transport container and environment have both humidity sensor embedded, it is not possible to make a statement about neither absolute nor relative humidity, because both sensors will most likely have different sensitivity, different resolutions or only a different mounting. Because it not feasible to transfer information about how to interface the data with the sensor network message, only domain knowledge helps us to process it.

This is why we propose to externalize the message decoders and to make interface descriptions explicit by introducing a service view on sensor network functionality. Services are self-descriptive, i.e. they provide information by publishing a service description. The concrete technology used to implement functionality is hidden behind that interface. Service interfaces provide the client with typed and attributed data. Because service oriented architectures have a standardized, uniform interface to all services, only one message decoder is needed to process the message. Using standardized service interfaces also allows seamless integration with other application frameworks.

2.2.3 Interoperability on Overlay Networks

Up to this point we have assumed that supporting IP networks alone gives us ubiquity in terms of connectivity. However, in existing systems network topologies build technical barriers between the peers. The Internet structures the network by their providers and partitions domains by firewalls and NAT gateways. Especially in production sites the internal networks are highly restrictive and difficult to manage.

Furthermore, because WSNs are non-IP networks a client will have problems efficiently routing packets to a specified sensor node. Because of different ID schemes we would need address translation mechanisms for mapping them to IP addresses.

The centralized architecture of Internet services like DNS or UDDI repositories introduces unnecessary single point of failure and leads to scalability problems [8]. Exposing a large amount of wireless sensor nodes to this network will only amplify these problems.

We propose to create a virtual overlay network that reflects topology and properties of WSN communication instead of using plain IP technology for coupling remote sensor network sites. By using P2P messaging technology we are able to create a scalable virtual network on top of the Internet, while staying mostly independent of the physical network layout.

Consequentially, P2P networks can extend the principals of WSN addressing and routing mechanisms in order to seamlessly integrate local WSNs into IP networks as proposed in section 2.2.1. We use the service view proposed in section 2.2.2 in order to access the functionality from these overlay networks.

2.2.4 Confidentiality

Often practical security concerns hinder the application of P2P systems. Adding security features such as access control to high-level functional service interfaces, however, lead to a high runtime and maintenance overhead. In contrast to that specifying sensor data that can pass the security is relatively easy.

Therefore we propose a confidentiality control mechanism that can directly work on the data. Requests for data forwarding are checked against a local policy. This way a seamless real-time temperature monitoring could be guaranteed from factory A to factory B, without revealing movement information.

3. Architecture

In this section we will combine the proposals from the previous section in a single Plug and Play service oriented architecture (SOA). Our middleware approach depicted in Figure 2 consists of three main parts that help to interconnect sensor node and client application. The P2P Bridge interconnects the WSN to the peer-to-peer messaging network, the P2P substrate that creates an overlay network for all WSNs and the P2P UPnP Gateway for providing service interfaces to the client application.

3.1. Plug and Play Interfaces

A service oriented and lightweight architecture can be implemented using UPnP technologies. This technology also allows us to incorporate other embedded devices into our application as they often already provide UPnP functionality. Low cost, energy efficient sensor nodes will probably not have the capabilities to support to UPnP natively. Furthermore, native UPnP support

would have other serious disadvantages for sensor networks. Instead of running UPnP on a sensor node a gateway translates RPC and events to sensor network messages and vice versa. For this reason we instantiate a local UPnP Proxy that implements the UPnP protocol stack.

3.2. Extending P2P Communication

The current UPnP architecture is designed for LAN environment. As a consequence, it does not consider the global use that requires global wide service discovery, efficient support for firewall traversal, and loosely coupled service providers. UPnP uses SSDP a UDP multicast for service discovery and lightweight HTTP/SOAP servers as service providers. These protocols assume direct reachability of all peers and only work in friendly network environment. For this purpose, we propose installing a P2P Bridge at the remote WSN. This bridging component exposes the WSN to a P2P network and enables the P2P UPnP Gateway to discover remote sensor nodes through the P2P substrate and to instantiate UPnP Proxies for them to ensure client connectivity.

Sensor nodes have many semantics to describe themselves, not only unique ID or address in the network layer, but also their location, the type of artefact to which they are attached, the type of the sensors they host, etc. All these attributes can be utilized as addressing schemes. However, IP networks do not support this kind of addressing. In our architecture addressing the sensor node according to their semantics is achieved by using the P2P discovery functionality.

In the state of art P2P protocols, DHTs (Distributed Hash Tables) (e.g. [7]) are used to discover resources on the P2P substrate by querying for their descriptions. This kind of distributed discovery eliminates the need for centralized services and has better performance than multicast.

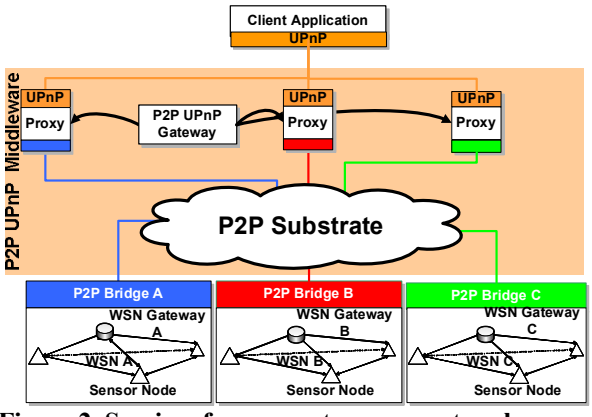

Figure 2. Services from remote sensor networks

4. Implementation

In the last section we have outlined the architecture that enables us to couple services provided by remote sensor networks over an IP network like the Internet. We implemented the P2P Bridge and the P2P UPnP Gateway in Java 1.5 with JXTA 2.3.2 as a P2P platform and the Cyberlink UPnP Stack. We applied Particle computers as sensor nodes. Within the WSN they transmit their data packets utilizing a low power radio protocol.

4.1. P2P Bridge

As a gateway to the WSN we install the P2P Bridge in order to expose the WSN to an IP network. The P2P Bridge receives the packets and analyzes the encapsulated sensing data to extract the attributes of the sensor node, ID, location ID and type of artifact, by using libparticle API to communicate with Particle computers. These attributes are treated as the semantics of the sensor node and published to the P2P substrate as an advertisement. An associated pipe ID included in the advertisement serves as an identifier to establish the messaging connection with other peers. When a pipe is established, the P2P Bridge forwards the received sensing data in the form of XML document. In JXTA P2P substrate, a party of peers can organize a Peer Group that requires an authentication to participate. It helps to keep a basic confidential of the party.

When a peer makes a Discovery Query to identify a set of sensor nodes, it utilizes attributes of the sensor nodes as a query key. The query is resolved by the discovery functionality of the P2P substrate. If an advertisement published by the P2P Bridge matches the query the peer has discovered a sensor node. The peer can now access the node by the returned pipe ID. For instance, a query that contains the type of a sensor, e.g. “humidity”, as attribute can discover sensor nodes hosting a humidity sensor.

Additionally, the peer can retrieve data by publishing a Forwarding Request without establishing a connection explicitly. The request specifies the types of data for which the peer subscribes. This function is suitable for retrieving a service specific data from a number of sensor nodes, e.g. retrieving high temperature alarms. All P2P Bridges that discover such a request will forward the sensing data that matches the set of attributes to the peer as well as the local confidentiality policy. Thus the confidentially policy is nothing more than a filter that needs to be applied to all outgoing connection.

4.2. P2P UPnP Gateway

As illustrated in Figure 2, the P2P UPnP Gateway is responsible for providing the client application with an interface to the WSN Services offered from WSNs over

the P2P network. WSN Service Descriptions that describe WSN Services utilizing the functionalities of sensor networks are provided to the P2P UPnP Gateway from a WSN service provider. The P2P UPnP Gateway can now instantiate a UPnP Proxy for each WSN Service offered by the WSN Service Description. If an UPnP Proxy is started, it installs an appropriate Forwarding Requests to discover the data requested by the WSN Service. By analyzing the WSN Service Description we restrict these Forwarding Requests to the minimum of message types needed to locally replicate the service. It significantly reduced the network traffic as only actually interfaced data is published on the P2P network.

In the client network the UPnP Proxy transforms the service interfaces in RPC and state variable oriented UPnP interfaces in order to support intuitive client accessibility. Mapping asynchronous messages to local state variable allows us to create robust interface to otherwise unreliable sensor node communication. By replicating the service state locally in an UPnP Proxy and interpreting messages as remote state changes, we totally decouple the service provider from the consumer. Currently the message parser/generator supports tuple and XML oriented message formats, however, can be easily extended to support another kind of encoding.

4.2.1 Service Discovery

The client can request local service instantiation by calling the Lifecycle Management Service of the P2P UPnP Gateway Device by using UPnP. The gateway will then dynamically create UPnP Proxy for all discovered sensor nodes offering the requested WSN services. After the instantiation, local service discovery is completely handled by UPnP via SSDP. Additionally, a client can issue specific local discovery requests by SDDP URI searches. A client may for example query all devices with a temperature service available.

4.2.2 Service Interaction

For interaction with a specific service **Error! Reference source not found.** shows in more detail the components and the path of a message when processed by our system. The WSN Service Description used to instantiate an UPnP Proxy is an extension of the UPnP Service description. In addition to the interface description it contains a Message Transformation descriptions for every function offered by the service. These transformations formally specify the encoding used by the sensor network.

The encoding is currently specified by message templates containing wildcards and don't care fields. The wildcards are linked to local state variables. On each incoming message the message is checked against fixed fields in each template and on match the local state variable is updated with the data underneath the wildcards. The template works also the other way around. When a state variable is changed by the client by calling a service function with input arguments, the scheme is reverted and the arguments are filled into the wildcards of the message template obeying the specified encoding and the message is send.

By defining blocking and non-blocking behavior for function results we support four basic interaction methods:

- Send (non-blocking)
- Receive (non-blocking)
- Call (blocking)
- Callback (non-blocking)

Send, receive and call are mapped to SOAP RPCs whereas for callback the client registers callback URL for GENA NOTIFY events.

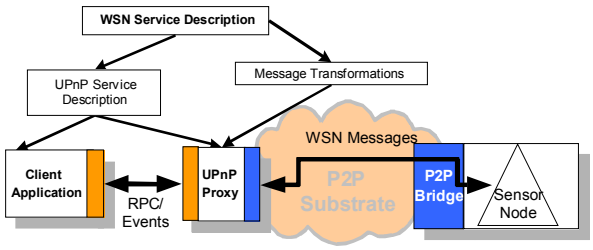

Figure 3. UPnP Proxy for WSN Service

5. Application/Evaluation

For the scalability, our largest trial of the P2P bridging technology employed 312 uPart Particle Computers simultaneously. The data was transferred via a P2P network between Tokyo, Japan, and Karlsruhe, Germany, traversing multiple firewalls.

In the following we analyze the efficiency of the Forwarding Request of P2P Bridge relating to this setup. The total length of the sensing data from the uPart is 32 Byte, while 24 Byte is redundant information to compensate for packet loss in WSN. The P2P Bridge in this environment each sensing values are embedded in XML. We encode each byte of the sensing data by a string representation (n*3 byte) plus an XML tag (67 byte). 536 byte strings are added permanently for node and location ID, sequence numbers, etc. Without applying selective data routing the Forwarding Request triggers 886 Byte to be sent with all kinds of sensor values. In contrast to this, sending only a subscribed sensing value spares about 30% of network load. The resulting message sizes of the information as it is passed through our system are summarized in **Error! Reference source not found.**.

There are many efforts to evaluate the general performance of the underlying JXTA (e.g. [9]). From these efforts, JXTA pipe messaging was approved to achieve almost the same throughput as plain socket based com-

munication. However, it has an overhead caused by applying XML, the latency becomes worse than the plain socket, and it implies that the message reduction by Forwarding Request will be important. Also because we can use efficient XML to XML transformations in the UPnP Proxy the overhead is acceptable.

To measure UPnP event notification, we used a 1.4 GHz Intel Pentium CPU running a 2.6 Linux kernel. The time for the message transformation is below 1 millisecond (ms). However, time for client notification strongly depends on the TCP connection setup and the control point response. The overall delay for passing event notification from the P2P network through the UPnP Proxy to the client is only around 10 ms. However, because of in order and acknowledged execution of the notification by the UPnP protocol stack this can add up linearly if the client increases the service subscription (see Figure 4). Also a high event rate will also lead to cascading response times. For this reason we can support an overall event rate of 90 events/s for every UPnP client.

From the viewpoint of discovery functionality, JXTA keeps a reasonable response time even the number of peer is increased. However, we must assume that the number of peers and published advertisement will further increase if many WSN are interconnected. Although we believe that our system can scale beyond the point that has been already evaluated, we need further evaluation from large-scale application trials.

Table 1 Message sizes for temperature service

Message type	payload	packets
Sensor network message	32 byte	1
JXTA Message	606 byte	2
UPnP SOAP RPC	959 byte	5
UPnP GENA Event	314 byte	5

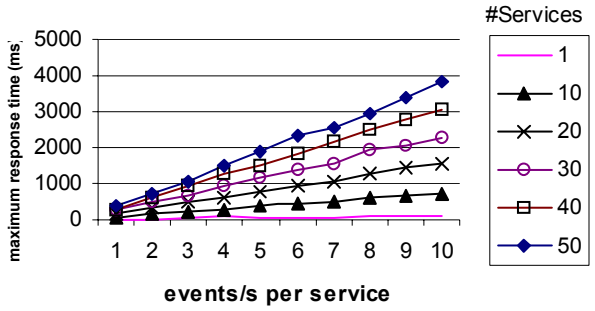

Figure 4 Delay of UPnP event notification

6. Conclusion and Future Work

Starting from an application in the area of distributed manufacturing we derived the requirements for the sharing of sensor networks as new resources in this domain. The necessary abstraction was implemented using the service oriented UPnP standard and a template based message transformation between UPnP and WSNs. In combination with P2P networks our approach enabled the usage of UPnP in wide area networks such as the Internet. Our trial with 300+ sensor nodes organized in a P2P network spanned between continents is an excellent proof-of-concept. Future work includes research in more powerful service abstractions than UPnP. Another focus is the investigation of automated service composition for WSNs, where our approach has delivered basic contributions. Finally, we will approach the integration in a real world distributed manufacturing system for logistic processes.

7. Acknowledgements

The work presented in this paper was partially funded by the EC through the project CoBIs (Collaborative Business Items) under contract no. 4270. In addition, we appreciate the related persons in KDDI Inc. who gave us the opportunity to do this collaborative research.

8. References

[1] P. Sousa, N. Silva, T. Heikkila, M Kollingbaum, P. Valckenaers, "Aspects of Co-operation in Distributed Manufacturing Systems," Proceedings of the 2nd Workshop on Intelligent Manufacturing Systems, Leuven, NL, 1999

[2] J.Heidemann, R.Govindan, "An Overview of Embedded Sensor Networks," Technical Report ISI-TR-2004-594, USC/Information Sciences Institute, 2004.

[3] C.Y. Huang, C. Pattinson., "Using Mobile Agent Techniques for Distributed Manufacturing Network Management," Proceedings of the 2nd Annual Symposium of Postgraduate Networking Conference (PGNET'01), Liverpool, UK, 2001.

[4] C. Decker, A. Krohn, M. Beigl, T. Zimmer, "The Particle Computer System," Proceedings of the ACM/IEEE IPSN 05, Los Angeles, USA, 2005.

[5] C. Decker, M. Beigl, A. Krohn, U. Kubach, P. Robinson, "eSeal - A System for Enhanced Electronic Assertion of Authenticity and Integrity of Sealed Items," Proceedings of the Pervasive Computing, Wien, Austria, 2004.

[6] Universal Plug-and-Play Architecture, Microsoft, 1999, http:///www.upnp.org/download/UPnPDA10_20000613.htm

[7] A. Rowstron, P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms, 2001.

[8] B. Traversat, M. Abdelaziz, M. Duigou, J. C. Hugly, E.Pouyoul, B. Yeager, "Project JXTA Virtual Network," SUN Microsystems, 2002.

[9] G. Antoniu, P. Hatcher, M. Jan, D. Noblet, "Performance Evaluation of JXTA Communication Layers," Proceedings of the Fifth International Workshop on Global and Peer-to-Peer Computing, 2005.