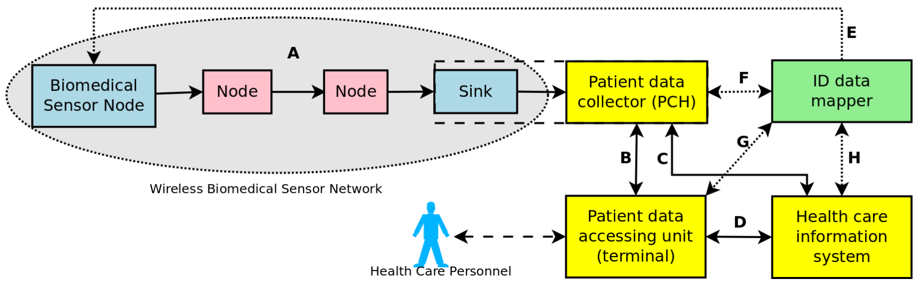
Norsk Regnesentral
NORWEGIAN COMPUTING CENTER

# Medical Digital Items for Use in Patient Monitoring Systems


A
Biomedical Sensor Node
Node
Node
Sink
Wireless Biomedical Sensor Network
Patient data collector (PCH)
F
ID data mapper
E
B
C
G
H
Patient data accessing unit (terminal)
D
Health care information system
Health Care Personnel

Note

| | |
|---|---|
| Note no | DART/13/10 |
| Authors | Wolfgang Leister |
| | Trenton Schulz |
| Date | December 15, 2010 |

## The authors

**Wolfgang Leister**, Dr. rer.nat., is a chief research scientist at Norsk Regnesentral, with research interests in multimedia, computer graphics, computer and sensor networks, health care applications, mobile systems, and free software.

**Trenton Schulz**, M.Sc., is a research scientist at Norsk Regnesentral. His interests include human-computer interaction, e-Inclusion, universal design, mobile systems, software engineering, and open-source software. In his previous career Trenton worked as a software engineer at Nokia.

## Acknowledgements

This research is funded by the SAMPOS project in the VERDIKT framework of the Norwegian Research Council. The authors thank Thor Kristoffersen for initial work with applying Exificient to the medical digital items, and Ilangko Balasingham for discussions and providing medical data for testing our ideas.

## Norwegian Computing Center

| | |
|---|---|
| **Title** | **Medical Digital Items for Use in Patient Monitoring Systems** |
| **Authors** | **Wolfgang Leister, Trenton Schulz** |
| Quality assurance | Knut Holmqvist |
| Date | December 15, 2010 |
| Publication number | DART/13/10 |

## Abstract

We show how to design MPEG-21-based Medical Digital Items for use in patient monitoring systems. The use of XML in biomedical sensor networks is challenging due to the size of XML documents and the resource constraints on sensors. We show how to use alternative compressed and streaming representations of the medical data in sensor networks to overcome these problems. We also performed initial measurements in a testbed specifically designed for this purpose.

| | |
|---|---|
| Keywords | Medical Digital Item, MPEG-21, Patient Monitoring Systems, Wireless Sensor Networks, Health Care |
| Target group | Researcher community, Health Care Enterprises |
| Availability | Open |
| Project | SAMPOS |
| Project number | 320365 / NFR:176875/S10 |
| Research field | Multimedia, Networks, Security |
| Number of pages | 15 |
| © Copyright | Norwegian Computing Center |

## 1 Introduction

Patient monitoring systems are a major data source in health care environments. It is important that patient monitoring systems maintain a certain level of availability, Quality of Service (QoS), and that they are secure and protect the privacy of the patient. Previously, we have analysed the security and privacy for patient monitoring systems with an emphasis on wireless sensor networks (Leister et al., 2009), and suggested a framework for providing privacy, security, adaptation and QoS in patient monitoring systems that makes use of medical digital items (MDI), and builds on relevant parts of the ISO standard MPEG-21 (Burnett et al., 2006). However, our architecture has not been deployed yet since several practical issues need to be addressed.

Leister et al. (2010) divided a patient monitoring system into four generic levels: (0) the patient; (I) the personal sensor network; (II) devices in the closer environment following several scenarios; and (III) the health care information system. Aligned with this, a generic system model is developed, here shown in Figure 1. Entities and communication channels in a patient monitoring system are characterised. We show the communication Channel A for the personal sensor network, Channels B, C, and D for information channels in the other layers, while Channels E to H denote channels used by the ID data mapper which represents a virtual security functionality rather than a physical entity. Later, in Section 3 we show how these channels are used.

In a practical deployment, biomedical sensors are (possibly wirelessly) connected to a bedside patient cluster head (PCH) acting as a patient data collector, which in turn is connected to the hospital infrastructure, or directly to a terminal enabled to access medical digital items at which medical personnel or the patient can access the patient data. Our proposed architecture supports content adaptation according to terminal and network capabilities, QoS and energy efficiency, and several types of session mobility.

The contribution of this paper is to show how our architecture using the medical digital items can be deployed. We elaborate which information is exchanged in the channels for the different phases of operation, and how this information is exchanged. The medical digital items must be designed carefully to meet the security requirements, and to be suitable for all involved devices. Since the exchange of XML documents defined by MPEG-21 is not viable for resource-limited devices, we need to transfer these data in a form that requires less resources. We also need to make estimates about the size of the MDI to see whether we meet limitations in the used transfer technologies.

Our question is, given the generic system model and the idea of using the MDI, how can medical data in patient monitoring systems flow securely? To answer this question we need to optimise the MDI into the form of the $\mu$MDI, for which we evaluate several options in the form of measurements in a testbed.

In the next sections we give an overview of relevant standards (Section 2) before presenting how channels are used in our architecture (Section 3). The design and implementation of the MDIs in a testbed is explained (Section 4), before concluding (Section 5).

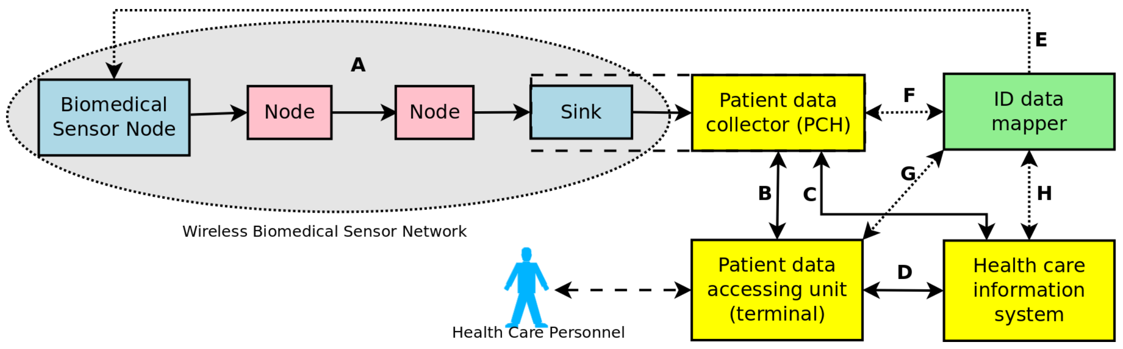

A
Biomedical Sensor Node
Node
Node
Sink
Wireless Biomedical Sensor Network
Patient data collector (PCH)
F
ID data mapper
E
B
C
G
H
Patient data accessing unit (terminal)
D
Health care information system
Health Care Personnel

Figure 1. Generic system model with Channel A shown in detail.

## 2 Background

Our architecture uses MPEG-21 to implement the generic system model. For the deployment, we apply the XML-based standard MPEG-21 to the medical data in patient monitoring systems, and also see into binary XML, and digital item streaming.

### 2.1 MPEG 21 and Medical Digital Items

MPEG-21 (Burnett et al., 2006) is an XML-based standard published as ISO/IEC 21000 by the International Standardisation Organisation (ISO) (International Standards Organisation, 2004). It offers features such as adaptation and security for network and user. Note that MPEG-21 does not implement the security measures, but encapsulates the data in an appropriate manner. One benefit of basing our proposed architecture on an international standard is that it makes our architecture both vendor neutral and technology independent.

In MPEG-21 data are encapsulated into digital items (DI). We developed a specific version of the DI to cover the use in a health care environment, defining the medical digital item (MDI), into which the mechanisms for adaptation and security are encapsulated. For devices with restricted capabilities we define a specific version of the MDI, the so-called $\mu$MDI which represents a compressed version of the MDI, where the XML syntax is not employed in order to save battery capacity and processing power.

The medical data being transferred in a patient monitoring system form a multimedia data stream, where data are packeted together with meta-data, and transferred over the channels outlined in the generic system model. To transfer the information, we encapsulate the data into a format that is streamable; our solution was to use Part 18 of MPEG-21 – Digital Item Streaming (International Standards Organisation, 2007c).

### 2.2 Binary XML

Binary XML refers to any specification which defines the compact representation of the Extensible Markup Language (XML) (Bray et al., 2006) in a binary format. While there are several competing formats, none has been accepted as a de-facto standard yet. Using a binary XML format reduces the size, and eases parsing of the documents at the cost of human-readability. Binary XML is used typically in applications where performance or resource limitations apply. For an overview, we do not consider traditional compression methods applied to XML documents, such as using gzip, or an existing standard like ASN.1.

The ISO and the International Telecommunications Union (ITU) published the Fast Infoset standard (International Standards Organisation, 2007b). The World Wide Web Consortium (W3C) defined the Efficient XML Interchange (EXI) format specification (Peintner and Pericas-Geertsen, 2009; Schneider and Kamiya, 2009). Another ISO standard is ISO/IEC 23001-1 (International Standards Organisation, 2006), also known as Binary MPEG format for XML (BiM). BiM is used by many ETSI standards for digital and mobile television. WAP Binary XML (WBXML) (Wireless Application Protocol Forum, 2001) is defined by the Open Mobile Alliance (OMA) for applications using the Wireless Application Protocol (WAP), and has also been proposed by the W3C (Martin and Jano, 1999). There are more binary XML formats available, designed for specific application domains. Below, we look closer into the two candidate technologies for our architecture, BiM and EXI.

BiM is part of the MPEG-7 standard and is also part of MPEG-21 (Part 16). It specifies a general method for compressing and decompressing XML documents for efficient transport and storage. It does this by examining the schemas for the document and using that information to create a separate decoder header that is used for encoding and decoding documents that use those schemas. The encoder uses this information to encode and partition the document into fragments. These fragments are then sent to the decoder. The decoder reassembles the fragments into a semantically similar document.

EXI is defined by the W3C. Its purpose is to have a "...very compact, high performance XML representation that was designed to work well for a broad range of applications," (Schneider and Kamiya, 2009). Currently, it is a W3C Candidate Recommendation. EXI is *schema informed*, which means that it can use the schema to create a more efficient document, but a schema is not necessary. EXI is compatible with other documents at the XML Information Set level, but not at the XML syntax level. This means that it can inter-operate with other XML documents at the Information Set level (Schneider and Kamiya, 2009). EXI does not have explicit support for fragmenting; it depends on other XML technologies for this.

## 2.3 EXI and BiM Comparison

While both BiM and EXI compress XML documents, they work in different ways. The EXI standard aims at compressing an entire document, schemas included, and transferring it in a compressed form as one stream. BiM starts by sending the compressed schemas as a separate file – called the the decoder config – and then sending the compressed XML, potentially as fragments. One advantage to this approach is that the decoder config only needs to be created and sent once (or stored ahead of time). It also means that the same decoder config can be used to decode any BiM compressed document that uses this set of schemas. If additional schemas are added later, a new decoder config needs to be generated. This does not make the current decoder useless: it just cannot decode elements that use the new schema.

BiM and EXI deal with updating documents differently. BiM defines specific algorithms and structures for dealing with document fragments. This includes specifying where the fragment update should be and what needs to updated, added, or removed. The result is that the majority of the work with regards to updates is handled by the implementation itself. It is only a matter of transferring the fragment updates over to the implementation.

EXI defines a fragment of a document, but there are no specifics in how a potential fragment could be used. It is up to the receiver to determine how the fragment should fit in the existing document. Even though EXI does not have its own facilities for document updates, other XML-transforming technologies – such as XSLT or XQUERY – could be used together with EXI. There is added complexity on the side of the client and server, but it is an option.

## 2.4 Available Implementations

The initial plan was to use BiM since we could split a document into fragments, compress these, and send them all with one tool. EXI would only give us the compression. Ultimately, the final decision came down to the quality of implementations that were available since we did not have time nor resources to create our own implementation. The BiM implementation (Technische Universität München, 2008) seems to be incomplete, is not documented, and has many bugs. Initially, it would not encode our $\mu$MDI document at all. After a lot of tracing and fixing, we were able to get it to encode a document similar to $\mu$MDI, but it then failed to decode it correctly. It also seems to be missing the code for partitioning the document into fragments and reassembling it: the result is always a file, not a set of fragments. Since BiM is a standard, one can expect that other implementations exist that address these issues. But, this was the only available implementation that we could find for our experiments.

By contrast, the EXI implementation (EXIficient) is available as an open source project (Peintner, 2010). While it is missing some documentation, it is actively developed and seems to handle all the situations we have given it. Since EXI has no built in streaming, we needed a method for streaming the items. This was solved using Digital Item Streaming (DIS; see Section 2.5).

ISO has an implementation (International Standards Organisation, 2007a) that works well and is straightforward to extend. However, one issue with DIS is that it only works with a "completed" document. That is, one cannot add siblings to the document after having sent over the basic skeleton. Therefore, future packets must be children of the current elements.

## 2.5 Digital Item Streaming

MPEG-21 Part 18 Digital Item Streaming (DIS) (International Standards Organisation, 2007c) works by specifying a separate XML document that defines the Bitstream Binding

Language (BBL) for a document. This BBL specifies the packets or packet streams for the document, when these packets should be sent, and how the source document should be divided up. The BBL is then used by software to divide the document up and send it over arbitrary protocols. DIS also allows specifying constraints for packet streams (such as number of items, or packet size), but the standard indicates that constraints *will* be broken in order to send a packet.

# 3 Usage of MDI in the architecture

As outlined by Leister et al. (2009, p. 24), the life cycle of a biomedical sensor node in a patient monitoring system consists of the four phases (*1*) initialisation (sensor node receives appropriate software, keys, and schemas); (*2*) deployment (establish relationship between patient and sensor node); (*3*) operation (send medical data to the PCH); and (*4*) operation completion (invalidate the relationship between patient and sensor node). In order to use the $\mu$MDI, each of the above phases requires dedicated operations which will be elaborated in the following, and aligned with the channel letters used in Figure 1. Note that in the Phases (1), (2), and (4) the *ID data mapper* and the Channels E to H are involved, while the medical data use the Channels A to D in Phase (3).

## 3.1 Initialisation of the sensor

When a sensor node is initialised, it receives the appropriate software, identification credentials, keys, and schemas. All other involved entities, i.e., the PCH, the terminals, and the hospital infrastructure will also be prepared to receive the necessary credentials, yet not attached to a patient.

For Channel E the identity credentials are are transferred either *a*) in the form of data tables containing identities of the node and the data streams including the necessary keys and encryption schemes, or *b*) in the form of a compiled program, that contains the above information in encoded form. Note that for Channel E, the exchange of these data in the form of XML documents is not recommended if there are resource limitations. For the Channels F, G, and H the necessary data can be transferred, or be prepared for transfer, preferably in the form of XML documents. Note also, that in the initialisation phase no patient identities are involved.

## 3.2 Deployment Phase

When the sensor node is being deployed, it is attached to the patient. The sensor and stream identities are made known to the PDC. Sensor and stream identities are tied to the patient identity (accession number). For this, communication with the hospital infrastructure, or the accessing unit (terminal) via Channel C or B might be necessary to retrieve the patient identity, and the necessary security credentials. Note that the PDC needs to be authenticated, and the communication to and from the PDC needs to be secured by encryption.

## 3.3 Operation Phase

During the operation phase the medical data are transferred as $\mu$MDI using Channel A, and as MDI using Channels B, C, and D. The operations of Channel A are further elaborated later in Section 4.

## 3.4 Completion Phase

After the operation phase of a sensor node is completed, the PDC is supposed no longer to receive data from this node. Therefore, the credentials for the involved streams are invalidated; e.g., by deleting these credentials on the PCH. For this, no communication on Channel E is necessary; possibly, there is communication on Channels F to H to inform other entities that the operation phase has been completed.

# 4 Design and testing with MDI

The MDI that are sent during the operation phase need to be carefully designed in order to meet the security requirements, and also in order to conform to other constraints, e.g., packet size or resource limitations.

## 4.1 Design of the $\mu$MDI

A $\mu$MDI created from a sensor node contains the following data: *1*) sensor id; *2*) stream id; *3*) sequence number of packet; *4*) timestamp; *5*) type of sensor data; *6*) the sensor data; and possibly *7*) fields necessary for encryption. Meta-data describing the sensor data (not necessarily all meta-data in all packets; some of the meta-data which are constant could be sent in separate meta-data packets once in a while).

Examples of $\mu$MDI include:

- one measurement value from a device, but completed with all necessary data later (e.g., joined at the patient cluster head);
- many samples of values (e.g., sound data); the number of samples per packet needs to be determined, taking into consideration typical values for data types, such as EEG and ECG;
- a combination of measurements, such as measurements from several connected channels, or samples of one channel enriched with measurement of, e.g., a temperature value with a very low sampling rate.

The $\mu$MDI is enriched with the accession number (person identification), more meta-data about the media, possibly other necessary values in the PDC. We need to define which these are, and also make experiments with compression, etc. Note that the accession number and other data identifying a person are not to be sent from the sensor node to the PDC due to privacy reasons.

## 4.2 The size of the $\mu$MDI

The original Zigbee specification (ZigBee Standards Organisation, 2007) states that one packet can have the maximum size of 128 bytes. The first task is to figure out how much an $\mu$MDI document can be compressed.

Over several tests, we found that both EXI and BiM compare favourably to each other. Sometimes BiM won and sometimes EXI won. BiM would always create a 224 byte decoder config file that contains information about the schema and how it can be decompressed, but this is a one time operation assuming the namespaces used in the XML doesn't change. Therefore, it could be created ahead of time and shared among the nodes, so it would not need to be transferred.

Unfortunately, on their own, neither BiM nor EXI could compress items below the 128 byte limit. One issue could be that it is harder to compress an item in the Digital Item Definition Language (DIDL) than a custom XML type because DIDL tries to be generic and specify the attributes, whereas one could just make them normal nodes in the XML. For example, `<Item id="diastole">` is less compressible than `<diastole>`. It is an open question whether it would be an idea to leave MPEG-21 during the transmission, and, instead, find some ad-hoc replacement during the sensor network transmission.

On the other hand, BiM and EXI both offer ways of restructuring elements and support compression, so the issue with the DIDL XML might not be a problem. Both implementations can use the DEFLATE algorithm in the encoding to compress the document even more. For BiM this is for the encoding of string instead of UTF-8; for EXI it seems to be for the entire document. In one of our non-trivial tests, using DEFLATE with BiM resulted in a file the size of 120 bytes – under the 128 byte limit. With EXI it drops the document down to 126 bytes – also under the 128 byte limit. It should be noted that DEFLATE compression is labelled "experimental" in the BiM implementation that we used. Also, both the BiM and EXI encoders and decoders must be told about the alternate compression algorithm. This seems like it might be a solution if $\mu$MDI is mostly string-based. Yet, it should be noted that turning on the additional compression adds to CPU usage and can potentially lead to the payload being larger in some circumstances as seen in the Efficient XML Interchange Evaluation (Bournez, 2009).

Another option is to use a later version of the Zigbee specification, Zigbee 2007 and up, allow for message fragmentation and re-assembly, assuming that the buffer capacities of the devices are large enough (Daintree Networks, Inc., 2010). So, the fact that the document is larger than 128 bytes may not be a problem, assuming that the sensors have enough capacity for larger packets.

## 4.3 The Complete System

We combine $\mu$MDI, EXI, and DIS to form the following solution for a biomedical sensor network using MPEG-21. Since we are using DIS, which requires a completed document, we have expanded the $\mu$MDI to contain multiple entries. A portion of the new $\mu$MDI can be seen in Figure 2. This document is never sent as a complete document, but divided

```
<did:DIDL xmlns:did="urn:mpeg:mpeg21:2002:02-DIDL-NS"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:dii="urn:mpeg:mpeg21:2002:01-DII-NS"
  xsi:schemaLocation="urn:mpeg:mpeg21:2002:02-DIDL-NS schemas/didl.xsd
                      urn:mpeg:mpeg21:2002:01-DII-NS schemas/dii.xsd">
  <did:Container>
    <did:Item id="myitem">
      <did:Descriptor>
        <did:Statement mimeType="text/xml">
          <dii:Identifier>urn:grid:a1-abcde-9873216540-f</dii:Identifier>
        </did:Statement>
      </did:Descriptor>
      <did:Descriptor>
        <did:Statement mimeType="text/xml">
          <dii:Type>urn:sensor:bloodpressure</dii:Type>
        </did:Statement>
      </did:Descriptor>
      <did:Item>
         <did:Descriptor>
          <did:Statement mimeType="text/plain">20</did:Statement>
        </did:Descriptor>
        <did:Component>
          <did:Resource mimeType="text/plain">68.300</did:Resource>
        </did:Component>
      </did:Item>
      <did:Item>
      ...
```

Figure 2. Excerpt from a $\mu$MDI document adapted for Digital Item Streaming (DIS)

up using the BBL, and streamed via DIS (see Section 2.5). The base station has a skeleton of this document and each sensor can send specific items (such as the sensor identifier and its data) as separate packets. These packets are eventually received by the base station and are attached to the skeleton document. This document has a limited number of parts. After this document is finished, a new skeleton is added and the sensor can then retransmit its identity and continue sending its information. The base station can transfer this information on to the patient monitoring system.

The resulting packets should be small, but we can further reduce their size by using EXI and transmitting them. The base station will receive the items as an EXI stream and will apply them to the XML document. We found that when we combined the EXI packets with the deflate algorithm, the resulting size of the packets was around 91–93 bytes. Well under the 128 byte restriction.

The packets are sent from the biomedical sensor network to a PCH. The PCH takes the information and converts it from a $\mu$MDI to a real MDI, adds extra information (e.g., identifying information for the patient), and sends the information to other communication levels. This information is encrypted with strong encryption, ensuring that only medical staff with correct permissions may use it.

Another requirement to keep in mind is that both the BiM and the EXI implementations are available in Java, so they require a Java virtual machine on the device. This could bring in other performance considerations. We have only tried the implementations with Java Standard Edition and not any of the the mobile variants. Of course, there is nothing requiring Java in these implementations, so it would be possible to port the implementations to other languages given time.

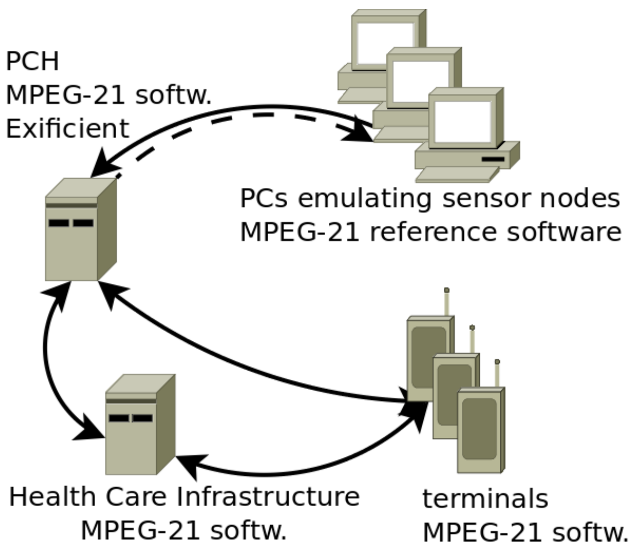

PCH
MPEG-21 softw.
Exificient
PCs emulating sensor nodes
MPEG-21 reference software
Health Care Infrastructure
MPEG-21 softw.
terminals
MPEG-21 softw.

Figure 3. The testbed for $\mu$MDI.

### 4.4 The Testbed for $\mu$MDI

Currently, we use a testbed to verify our results regarding the viability of our concept. Instead of implementing the software directly on a sensor node or other target devices, we implement the functionality as application programs on PCs, using the available implementations from the reference software. This gives us evidence about the size of the $\mu$MDI, and gives hints on how to implement the necessary schemas. However, the evidence that the software can be implemented on a sensor node can only be determined by estimating the complexity of the used algorithms.

The testbed, shown in Figure 3, consists of PCs emulating sensor nodes, using the MPEG-21 reference software and the EXIficient and DIS implementation. Another PC is used to emulate the receiver part, or PCH. Other nodes in the wireless sensor network are not considered in our abstraction, since these do not contribute to our evaluation.

Further PCs can be used to implement the health care infrastructure, and terminals to access the content. Note that many mobile devices are so powerful today that software implemented in Java can be run on these devices.

Using our testbed we can model and emulate the functionality of the entire patient monitoring network infrastructure. We are also able to experiment with different implementations of the MPEG-21 standard or other available software.

## 5 Conclusion

Transferring medical data, be it numbers, images, or video, can be seen as a special case of handling multimedia. Our architecture takes the MPEG-21 standard and deploys it to a new area of wireless sensor networks in a patient monitoring system. We have shown how to adapt the MPEG-21 to the resource constraints and security and privacy requirements in the health care area. We need to test our architecture in future scenarios to determine how well it works.

While MPEG-21's current Digital Items are too big to be used by small wireless sensors, our $\mu$MDI, combined with other XML technologies like compression and streaming, seem like a promising approach. We need to perform additional testing to see how our solution works with a real wireless sensor network and to determine if issues with reliability, security, and latency are also dealt with.

Our approach of using EXI and DIS was different than what we had expected at the beginning of the project, but it shows that combining these two standards together could be a viable replacement for BiM in a solution. Our solution also shows that the EXI standard is mature enough to be used for real-world scenarios. The lack of a quality BiM implementation was disappointing as we would have liked to have seen how the compression and streaming would have compared to our final EXI and DIS solution.

There is still more work that could be done. We would like to see how our architecture would interoperate with standards like ELIN (Christensen, 2009) or HL7 (Shaver, 2007). The ELIN and HL7 standards are used for communication in a hospital's infrastructure. Both ELIN and HL7 use XML for exchanging information, so it would be a matter of finding out what each of the standards expects and seeing how we could transform our MDIs to match, and vice-versa.

The IEEE 1451 (IEEE1451.0, 2007; IEEE1451.5, 2007) represents a family of smart transducer interface standards. These standards describe a set of open, network-independent communication interfaces for connecting transducers to microprocessors, instrumentation systems, and networks. The key feature of these standards is the definition of Transducer Electronic Data Sheets (TEDS) that store transducer identification, calibration, correction data, measurement range, and other relevant sensor node data. The IEEE 1451 can be relevant for the WSN part of a patient monitoring system, since the capabilities of sensor nodes and parts of the communication stack are represented in the standards. An initial analysis showed that our approach using MPEG-21 can be integrated with an IEEE 1451-based architecture. In IEEE 1451, security, adaptation, and application issues are not addressed. Therefore, our architecture using MPEG-21 can fill this gap, provided that a proper data exchange between these standards can be achieved.

We conclude that our approach using MDI and $\mu$MDI seems promising. Our implementation and measurements show that the approach is viable. Yet, several design decisions need to be made in order to create an optimal working system that is secure, provides the necessary QoS, adaptation, optimises energy consumption, and is reliable. Further integration towards the introduced standards in health care, and towards standards for sensors and transducers needs to be done.

## References

Bournez, C. (2009). Efficient XML interchange evaluation. Working draft, W3C. http://www.w3.org/TR/2009/WD-exi-evaluation-20090407/. 10

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F., and Cowan, J. (2006). Extensible markup language (XML) 1.1 (second edition). World Wide Web Consortium, Recommendation REC-xml11-20060816. 5

Burnett, I., Pereira, F., van de Walle, R., and Koenen, R., editors (2006). *The MPEG-21 Book*. John Wiley & Sons. ISBN 0-47001011-8. 4, 5

Christensen, T. (2009). Sluttraport ELIN. Sluttrapport, Den norske legeforening, Pb. 7000 St. Olavs plass, NO-0130 Oslo, Norway. In Norwegian. 13

Daintree Networks, Inc. (2010). Zigbee specification comparison matrix. Last accessed September 2, 2010. Available from: http://www.daintree.net/resources/spec-matrix.php [cited 2010-09-02]. 10

IEEE1451.0 (2007). IEEE standard for a smart transducer interface for sensors and actuators - common functions, communication protocols, and transducer electronic data sheet (TEDS) formats. *IEEE Std 1451.0-2007*, pages 1–335. 13

IEEE1451.5 (2007). IEEE standard for a smart transducer interface for sensors and actuators wireless communication protocols and transducer electronic data sheet (TEDS) formats. *IEEE Std 1451.5-2007*, pages C1–236. 13

International Standards Organisation (2004). Information technology—multimedia framework (MPEG-21—part 1: Vision, technologies and strategy. Technical Report ISO/IEC TR 21000-1. 5

International Standards Organisation (2006). ISO/IEC 23001-1: 2006. Information technology – MPEG systems technologies Part 1: Binary MPEG format for XML. Technical report. 6

International Standards Organisation (2007a). Directory listing for /publiclyavailablestandards/iso_iec_21000-8_2008_amd_1_2009_reference_software/. Last accessed September 8, 2010. Available from: http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_IEC_21000-8_2008_Amd_1_2009_Reference_Software/ [cited September 8, 2010]. 7

International Standards Organisation (2007b). Information technology—generic application of ASN.1: Fast Infoset. Technical Report ISO/IEC 24824-1: 2007 / ITU-T Rec X891. 6

International Standards Organisation (2007c). ISO/IEC 21000-18: 2007. Information technology – multimedia framework (MPEG-21) - Part 18: Digital Item Streaming. Technical report. 5, 7

Leister, W., Fretland, T., and Balasingham, I. (2009). Security and authentication architecture using MPEG-21 for wireless patient monitoring systems. *International Journal on Advances in Security*, 2(1):16–29. Available from: http://www.iariajournals.org/security/. 4, 8

Leister, W., Schulz, T., Lie, A., Grythe, K., and Balasingham, I. (2010). Quality of service, adaptation, and security provisioning in wireless patient monitoring systems. In *Biomedical Engineering, Trends, Researches and Technologies*. Intech. to appear. 4

Martin, B. and Jano, B. (1999). WAP binary XML content format. W3C note, W3C. http://www.w3.org/1999/06/NOTE-wbxml-19990624. 6

Peintner, D. (2010). EXIficient—open source implementation of the W3C Efficient XML Interchange (EXI) format. Last accessed September 8, 2010. Available from: http://exificient.sourceforge.net [cited September 8, 2010]. 7

Peintner, D. and Pericas-Geertsen, S. (2009). Efficient XML interchange (EXI) primer. W3C working draft, W3C. http://www.w3.org/TR/2009/WD-exi-primer-20091208/. 6

Schneider, J. and Kamiya, T. (2009). Efficient XML interchange (EXI) format 1.0. Candidate recommendation, W3C. http://www.w3.org/TR/2009/CR-exi-20091208/. 6

Shaver, D. (2007). HL7 101: A Beginner's Guide. *For the Record*, 19(1):22. 13

Technische Universität München (2007-2008). MPEG-7 reference SW. Last accessed September 8, 2010. Available from: http://www.lis.ei.tum.de/index.php?id=131 [cited September 8, 2010]. 7

Wireless Application Protocol Forum (2001). Binary XML Content Format Specification. Technical Report WAP-192-WBXML-20010725-a, Version 1.3. 6

ZigBee Standards Organisation (2007). ZigBee Specification. Technical Report 053474r17, ZigBee Alliance, Inc., 2400 Camino Ramon, Suite 375, San Ramon, CA 94583. 10