Andreas Reinhardt, Diego Costantini, Ralf Steinmetz: *Describing Packet Payload Structures using Lightweight Semantic Data Type Annotations.* In: Reiner Kolla (Hrsg.): Proceedings of the 9th GI/ITG KuVS Fachgespräch "Drahtlose Sensornetze", p. 25-28, September 2010.

Describing Packet Payload Structures using Lightweight Semantic Data Type Annotations

(Extended Abstract)

Andreas Reinhardt, Diego Costantini, Ralf Steinmetz
Multimedia Communications Lab, Technische Universität Darmstadt
Rundeturmstr. 10, 64283 Darmstadt, Germany
Email: {andreas.reinhardt, diego.costantini, ralf.steinmetz}@kom.tu-darmstadt.de

***Abstract*—In the majority of wireless sensor networks, packet structures are statically defined at design time. At runtime, sensed data is then inserted into the payload fields prior to packet transmission. While this is efficient in terms of the required processing, the packet structure cannot be modified during runtime. However, in certain situations the need for adaptation of the packets to new requirements arises, e.g. when the energy source approaches depletion and energy-hungry sensors are deactivated to extend the node lifetime. The countermeasure of defining a multitude of packet structures to encounter any possible situation is infeasible both in terms of efforts and resource consumption.**

To address this limitation, we propose the annotation of data fields in outgoing packets by identifiers indicating the contained data types, so that any node can send payloads with dynamically defined contents. The size increase incurred by the use of annotations for each payload field can however become significant as the annotations must be sufficiently expressive to uniquely describe the payload field. To keep this size increment small, we present a supplementary approach that assigns binary aliases for the used data type annotations, thus increasing the payload space available for application data. This is especially useful as payload sizes in sensor networks are generally limited by the radio protocol, and fragmentation is expensive in terms of the according energy requirement.

I. Introduction

A common characteristic in Wireless Sensor Network (*WSN*) deployments, e.g. in environmental monitoring settings like PermaSense [1] or GlacsWeb [2], is that the structures of radio packets used in these deployments have been designed in a static manner at design time. While such static packet structures eliminate the overhead of assembling packet contents dynamically before transmission, they also hamper the adaptation to the characteristics of the sensor devices. If nodes are fitted with multimodal sensing capabilities, the transmission interval is generally determined by the sensor with the smallest sampling interval. A second observation is that the use of convergecast routing algorithms, such as the Collection Tree Protocol [3], is prevalent. Packets with sensor data are forwarded along the branches of the routing tree to its root (the *sink* node), possibly relayed by a number of intermediate nodes. Although sensor data can be aggregated while being forwarded to the sink node ([4], [5]), statically defined packets impair the applicability of in-network data processing; data aggregation on intermediate nodes can only be performed to a limited extent when the representation of the results is limited to a set of pre-defined message structures.

To overcome these limitations, we propose to extend WSN applications to support the dynamic composition of packet payloads. Obviously, the definition of the packet structure must however be present at the receiver side to correctly interpret the contents of the packet. It can either be decoupled from the packet itself and transmitted in advance (such as done in ASN.1 [6]), or alternatively be provided inline with the packet payloads. As frequent changes to the packet structures (in some scenarios, each transmitted packet might be composed differently) would incur a great number of updates, we have chosen to provide the structure definition within the packets. As this is done on a per packet basis, the correct interpretation of packets is not impacted by packet losses.

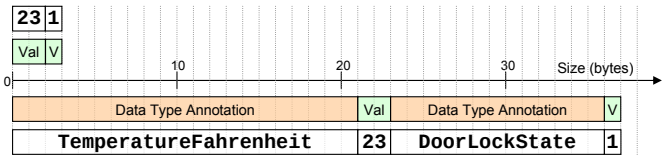

Fig. 1. Size of a packet without (top) and with (bottom) data type annotations

However the problem is that semantic data type annotations can lead to an increase of the packet size. In a simple example indicated in Fig. 1, only three bytes of payload values (`Val` and `V`) are transferred. The first field is 16 bits in size and carries a temperature reading, in the second field the state of a door lock is contained as boolean value. After the semantic data type annotation, the packet grows to 37 bytes in size. In this paper, we present an approach to reduce this overhead to a small fraction of the size shown, and highlight the benefits of dynamic packet composition. After presenting related work in Sec. II, we show a sample scenario benefitting from the use of annotations in Sec. III. We detail the use of semantic data type annotations in Sec. IV, and show in Sec. V how binary aliases can reduce the overhead introduced. We conclude this paper in Sec. VI, where we summarize our results and outline the next steps.

II. Related Work

The approach of introducing semantically annotated metadata in WSNs has been covered to some extent in related literature. The Open Geospatial Consortium (*OGC*) presents

an approach towards describing sensor devices in a semantic manner using the Sensor Model Language (*SensorML*) [7]. To enable the integration of such sensor systems into the semantic web, the Semantic Web Enablement (*SWE*) approach has been proposed in [8]. Both are however based on XML, and thus not sufficiently lightweight for application on embedded sensing devices. The Global Sensor Network (*GSN*) project [9] presents a middleware layer that abstracts all devices by *virtual sensors* and assigns semantic annotations. Inside the WSN, statically defined packets are used to transmit collected data.

Herzog et al. present the A3ME middleware in [10] with a special focus on the definition of sensor types and according messages. Their content representation is realized in a semantically annotated manner, with all pre-defined data types being stored in an ontology. Each time an unknown data type is present, it is indicated by an escape symbol followed by the complete description of the type. Embedded web servers present an emerging WSN application coping with variable packet payloads, enabled by applying TCP/IP to sensor networks [11]. In contrast to semantically defined packet payloads however, the application protocol defines how to decompose incoming messages and interpret their contents.

Maintaining all data types in an ontology represents the concept closest to our proposed use of dynamic payloads in WSNs. However, in contrast to a static data type ontology, we dedicatedly address possible dynamics in the network, i.e. new data types becoming present during runtime.

III. Illustration Scenario

The dynamic composition of packets is useful in many settings. In the remainder of this paper, let us e.g. envision a building surveillance sensor network, where each room is fitted with sensors configured to monitor a set of parameters. All nodes in the building form a routing tree and forward all their sensor readings to the root for processing and storage. An exemplary setup for one room is schematically shown in Fig. 2, integrating seven sensor nodes with twelve sensing modalities. While some of the sensors create continuous streams of data (such as noise level or brightness), others inherently generate events, e.g. indicating that a door or window has just been closed. Conventionally, all sensors would report their readings in a fixed interval, not taking the specific characteristics of the sensing devices into account.

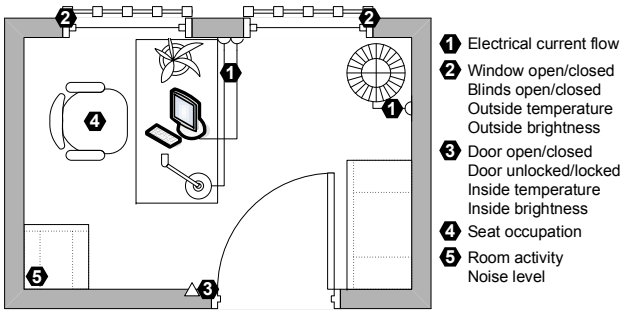

Fig. 2. Room monitoring scenario setup

Transferring each node's full set of sensor readings to the sink node, where it is stored for retrieval from third parties, in a statically defined packet leads to a complete update of the current building state, however it comes at the cost of packet payloads carrying the full set of sensor data. In contrast, confining packets to the relevant contents[1] through dynamic composition is a viable step towards preserving transmission energy. Also, when packets omit irrelevant fields from transmission, the resulting smaller payloads allow data fusion at intermediate nodes.

IV. Semantic Data Type Annotations

When packet payloads are no longer defined before the actual deployment phase, but instead composed during runtime depending on the availability of sensor data, the structural description of the packet payload must be present at the receiver to allow correct interpretation of the contents. Obviously, since providing a syntactic description of the data field type only (e.g. an unsigned integer of 16 bits length) would lead to ambiguities, semantic type annotations, such as `DoorLockState`, must be used supplementary. As semantic tags do neither imply their length nor the actual length of the data field, a length field and a separate syntactic type tag is used in combination with the semantic description.

The overall structure thus differs from the simplified form presented in Sec. I and is shown in Fig. 3. Every data field is now prefixed by both semantic and syntactic fields. First, the length of the semantic type tag is transmitted (entitled `L` in the figure), followed by the textual description of the semantic data type. Subsequently, the syntactic type of the following data is indicated (`T` in the figure, with types `S` indicating a 16 bit integer, while `B` refers to a boolean value). The syntactic type field is then followed by the actual sensor reading.

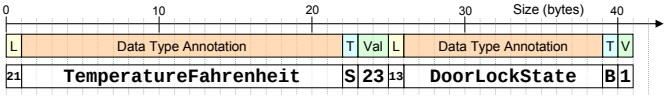

Fig. 3. Structure of a packet with syntactic and semantic data type annotations

As shown in the figure, we propose that metadata fields prefix the actual data field such that the received node can extract the type of data, the length of the corresponding field in the payload, and the value itself. After having provided a *meaning* to the value, all recipients who understand the given annotation tag can interpret the data accordingly. This enables data aggregation on nodes on the routing path, e.g. calculate the overall energy consumption in the entire building by multiplying `Voltage` and `Current` readings. Nodes to which the given data type is unknown can still forward the sensor data towards the sink. The length of the given payload field is defined by the syntactic description field, thus nodes can skip unknown types and proceed to the next field.

[1] As the process of determining relevant sensor data is beyond the scope of this paper, we assume that only significant changes to the sensor data (e.g. temperature shifts by at least one degree, or changes to the door lock state) necessitate the transmission to the building control server.

V. Consistent Assignment of Aliases

From the exemplary header structure shown in Fig. 3, it is clear that packet sizes are significantly increased when applying our proposed approach, as the semantic annotation of the data types requires the transmission of plaintext semantic data type descriptors. As long as readability by humans is intended, these verbose tags are well suited. However, in fully automated scenarios, valuable payload space can be saved by assigning aliases to the semantic data types. Instead of transmitting the plaintext value `TemperatureFahrenheit` with 21 characters, a field of a few bits in size can be used to represent the tag. In our design, we have used a field of one byte in size, which shares the same location as the length field. However, as the IEEE 802.15.4 standard [12] limits the packet size to 127 bytes, a maximum of seven bits can be required to represent the length of the semantic annotation. Using the remaining bit as an escape symbol, up to 128 data types may be present in the network. As a side effect, the constant length of the type field also obsoletes the corresponding length field.

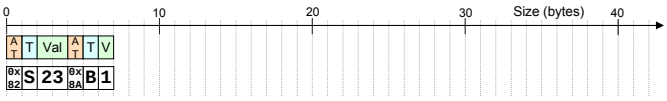

Fig. 4. Structure of a packet with semantic data type annotation aliases

Applying the to the previously shown packet structure, all semantic data type annotation fields are reduced to just a single byte. The resulting packet structure is visualized in Fig. 4 results, where a size reduction from 41 to 7 bytes can be seen (as compared to Fig. 3). Each entry is now only composed of the alias for the semantic type (`AT`, one byte), the syntactic field description (`T`, one byte), and the field itself (length inherently defined by the syntactic field description `T`).

In this presented approach, syntactic and semantic description elements are transferred separately from each other. If both are merged into a single descriptor field, another byte of payload size can be saved. Although feasible and beneficial in terms of size, we have deliberately decided to leave both fields uncoupled to allow other mechanisms to cater for efficient encoding of syntactic descriptors and data.

A. Definition of Aliases

The definition of aliases must take place in a coordinated order to avoid ambiguities. As the centralized sink node can provide the properties of transactional databases (i.e., atomicity, consistency, isolation, durability) best, we have imposed the tasks of assigning new aliases and storing a consistent mapping on this node.

To replace semantic data types by aliases on nodes, a local cache is implemented in all nodes. Whenever a sensor function returns data of a given semantic type, this local cache is checked for presence of an alias. In case an alias is known, the semantic type field in the outgoing packet is directly replaced by the shorthand notation. Otherwise, it is transmitted in plaintext, and a the creation of a new entry at the sink node is triggered, as described in the following section.

B. Adding Entries to the Dictionary

When binary shortcut forms for required data types are not known on the sensor nodes, they revert to the transmission of a plaintext semantic annotation, as shown in Fig. 3. Two possible situations may occur during the transport of the packet towards the sink:

1) An intermediate node has cached an according mapping between the plaintext annotation and the corresponding alias. In this case, the intermediate node replaces the field of the packet payload before relaying the data, aiming to minimize the size of the transmitted packet. In addition to forwarding a message towards the tree root, a notification message providing the corresponding mapping is also broadcast in the opposite direction (i.e., towards the origin of the packet), such that the sender as well as all nodes on the route may add the alias to their caches.
2) The data type has not yet been encountered in the network, and thus no mapping exists. In this case, the packet is forwarded to the sink node with plaintext data type annotation. There, a new entry is created and added to the mapping table, and the corresponding data type made known to the network through broadcasting.

In both cases, nodes should try to locally cache all annotation aliases for the data types they provide or consume, such that ideally, no plaintext annotations need to be transmitted after an initial setup phase. An excerpt of the mapping table generated in the exemplary room monitoring scenario depicted in Fig. 2 is shown in Table I.

C. Proactive Caching

In addition to storing mappings between for locally used semantic data type annotations only, nodes can also cache mappings for data types which are not used locally at all. While consuming additional RAM to store the mapping, this allows to resolve mappings closer to the node which has not yet stored the mapping for its data types (and thus sends the semantic data type annotation in plaintext). This way, a significant amount of traffic can be saved especially when the network is comprised of long routes. Caching mappings on intermediate nodes is possible, as binary aliases are only assigned by the central instance, i.e. the root node, so that no collisions can occur and a consistent state is guaranteed throughout the network runtime.

TABLE I
Aliases for Semantic Data Type Descriptors

Semantic Data Type	Alias
`RelativeHumidity`	`0x81`
`TemperatureFahrenheit`	`0x82`
`AccelerationX`	`0x83`
`WindowState`	`0x84`
`SwitchState`	`0x85`
`RelativeMotion`	`0x86`
`ElectricalVoltage`	`0x87`
`ElectricalCurrent`	`0x88`
`DoorState`	`0x89`
`DoorLockState`	`0x8A`

VI. Conclusion and Outlook

We have presented an approach to embed semantic data type annotations into packet payloads in WSNs. Opposed to static packet payload definitions, our solution allows to generate packets dynamically during runtime, and thus adapt to the characteristics of the attached sensor devices. Having shown that embedding semantic data types leads to significantly larger packet sizes, we have presented an approach to assign binary aliases to the data types, which are then used consistently throughout the network. Although the payload size is increased by two bytes per contained data field, the flexibility of dynamic packet composition allows to omit unchanged fields from transmission and enables more sophisticated in-network processing of data.

We dedicatedly focus on the efficient transfer of annotated data in wireless sensor networks and have therefore presented a mechanism to incorporate semantic elements into the network. It is essential to distinguish our design from related work where global ontologies describing sensor features are discussed. Any translation of the internally used data types to a global naming scheme is out of focus of our approach, but can be realized on a gateway device if necessary.

A. Outlook

As our next step, we target to complete the implementation of the presented mechanism for use on sensor nodes. We are planning to verify the effectiveness of our implementation on our TWiNS.KOM testbed [13], which integrates TelosB and SunSPOT devices. Special focus will hereby be put on scalable algorithms for the dissemination of aliases. In the long term, we are planning a real-world deployment of the resulting application on sensor nodes deployed in an office environment, like presented in Fig. 2, and target to investigate the achievable packet size and energy savings.

Acknowledgements

The authors would like to thank Parag S. Mogre and Stefan Schulte for the fruitful discussions and their contributions to this paper. This research has been supported by the German Federal Ministry of Education and Research (BMBF).

References

[1] J. Beutel, S. Gruber, A. Hasler, R. Lim, A. Meier, C. Plessl, I. Talzi, L. Thiele, C. Tschudin, M. Woehrle, and M. Yuecel, "PermaDAQ: A Scientific Instrument for Precision Sensing and Data Recovery in Environmental Extremes," in *Proceedings of the 8th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2009.

[2] K. Martinez, R. Ong, and J. Hart, "Glacsweb: A Sensor Network for Hostile Environments," in *Proceedings of the 1st IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, 2004.

[3] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection Tree Protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2009.

[4] B. Krishnamachari, D. Estrin, and S. Wicker, "The Impact of Data Aggregation in Wireless Sensor Networks," in *Proceedings of the International Workshop on Distributed Event-Based Systems (DEBS)*, 2002.

[5] T. Arici, B. Gedik, Y. Altunbasak, and L. Liu, "PINCO: A Pipelined In-Network COmpression Scheme for Data Collection in Wireless Sensor Networks," in *Proceedings of the 12th International Conference on Computer Communications and Networks (ICCCN)*, 2003.

[6] O. Dubuisson and P. Fouquart, *ASN.1: Communication Between Heterogeneous Systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001.

[7] M. Botts and A. Robin, "OpenGIS Sensor Model Language (SensorML) Implementation Specification," White Paper OGC 07-000, 2007.

[8] M. Botts, G. Percivall, C. Reed, and J. Davidson, "OGC Sensor Web Enablement: Overview and High Level Architecture," White Paper OGC 07-165, 2007.

[9] K. Aberer, M. Hauswirth, and A. Salehi, "The Global Sensor Networks middleware for efficient and flexible deployment and interconnection of sensor networks," EPFL, Tech. Rep., 2006. [Online]. Available: http://infoscience.epfl.ch/getfile.py?recid=83891

[10] A. Herzog, D. Jacobi, and A. Buchmann, "A3ME - An Agent-Based Middleware Approach for Mixed Mode Environments," in *Proceedings of the 2nd International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM)*, 2008.

[11] A. Dunkels, J. Alonso, and T. Voigt, "Making TCP/IP Viable for Wireless Sensor Networks," in *Work-in-Progress Session of the 1st European Workshop on Wireless Sensor Networks*, 2004.

[12] IEEE Std, "802.15.4 Part 15.4: Wireless medium access control (MAC) and Physical layer (PHY) specifications for low-rate wireless personal are networks (LR-WPANs)," 2006.

[13] A. Reinhardt, M. Kropff, M. Hollick, and R. Steinmetz, "Designing a Sensor Network Testbed for Smart Heterogeneous Applications," in *Proceedings of the 3rd IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp)*, 2008.