Proceedings of the SDR 11 Technical Conference and Product Exposition,

A Proposed API for the Information Plane of the WSN Integrated Technical Reference Model (I-TRM)

Babak D. Beheshti
Electrical & Computer Engineering Technology
New York Institute of Technology
Old Westbury, New York, USA
bbehesht@nyit.edu

Howard E. Michel
Electrical & Computer Engineering Department
University of Massachusetts Dartmouth
North Dartmouth, MA, USA
hmichel@umassd.edu

Abstract— The Integrated Technical Reference Model (I-TRM) for an autonomous Wireless Sensor Network (WSN) has been developed to be used as a guideline to develop a unified and standardized architecture for a diverse array of multi-platform WSNs. Based on the I-TRM proposed by Michel and Fortier, there are three planes to this reference model: The Information Plane, the Control Plane and the Behavior Plane. This reference model lays out a detailed layered model with functional description of each layer described in general terms. The Information Plane puts forward the information processing side of the system. The main focus is on data collection, information aggregation, knowledge generation and presentation. It shows how data is transformed into knowledge.

This paper presents the follow up research performed on this I-TRM, by providing a platform independent API to aid designers of WSNs to develop a codified implementation of WSNs. The API has been implemented using nesC in a TinyOS environment, running on the Berkeley Motes.

Keywords-component; wireless sensor networks; reference model; API

I. INTRODUCTION

WSNs are networks of tens, hundreds or thousands of "Sensor Nodes". Sensor Nodes are miniature size devices that are equipped with one or more sensors, measuring environmental elements such as temperature, light, vibration, humidity etc. Integrated with low cost and low power wireless communication modules, Sensor Nodes are designed to be limited in storage, computational resources and total energy available. A common sensor node platform used in the research community is the Crossbow Mote, shown in Figure 1. These low cost nodes can be strewn, or carefully placed in a deployment zone, allowing for collection, aggregation and communication of environmental data between themselves and ultimately to a user station. Sample applications are environmental monitoring in buildings, forest fires, volcanoes, battlegrounds, storms, underwater and other hostile environments.

WSNs are rapidly expanding in deployment in an astounding variety of applications. The effort involved in developing and productizing non-reusable code for WSNs that are based on a multitude of hardware platforms, operating systems and models is non-trivial and has lead researchers to find solutions to standardize the design philosophy and the design approach of WSNs.

Figure 1 - The MicaZ Crossbow Berkely Mote

The I-TRM proposed by Michel and Fortier attempts to standardize the structure of applications running on a sensor network based on a layered model. In this paper we specify the information face of this model in further detail by augmenting to its layer based definition, a set of API that can be used as a programmer's guideline for implementing these layers.

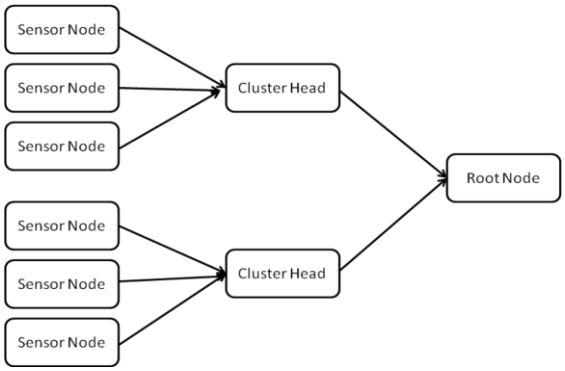

Figure 2 - A Possible WSN Hierarchy

II. PROBLEM DEFINITION

The concept of an integrated TRM for WSNs is a novel idea introduced by [3]. No other comprehensive WSN reference model has been proposed by others. This novel I-TRM, however has been so far defined in terms of generic responsibilities of individual layers and the overall relationship of the three faces of the model. No work has been done to take

the I-TRM to the next level of specifity and to propose a reference API for all appropriate layers.

The research work proposed here is to build on the foundations of the I-TRM, as laid out in [1] and [2], but to bring the existing foundations to a complete framework, along with a real system implementation of the framework. The functional decomposition of each layer and the interlayer interaction needs to be developed and specified. Based on the current work published in the literature, an API for the layered I-TRM model is to be developed, as well as inner workings of each layer is to be sufficiently outlined. This API, while generic and expandable, must contain sufficient level of specification in order to make it directly implementable in a test platform.

Once these technical specifications are complete, an embodiment of them will be implemented in a sample WSN. The current plan is to implement the layers 1 and 2 of the I-TRM, in the Micaz motes by utilizing and expanding the current TinyOS infrastructure and programming paradigm already in place. These motes will constitute the "Sensor Nodes". Similarly, layers 3 through 6 are to be implemented in a combination of a mote and a laptop. This would constitute the "Root Node" of the WSN.

The main objectives of this research project are as follows:

1. Based on the current state of the art in the literature, define an API definition for all relevant layers of the I-TRM. It is understood that while this API might not address all possible implementations of the I-TRM across the spectrum of applications, it should provide for a guideline for a programmer to be able to implement physical WSN utilizing the I-TRM.
2. In order to validate the proposed API (in the context of the I-TRM), develop a full WSN system, demonstrating a significant subset of the API's offerings. This WSN will have the following components:
    a. Sensor node source code developed for a sample WSN system based on a commercially available sensor node platform, implementing the proposed I-TRM API.
    b. Source code developed for the Root Node, implementing the proposed I-TRM API.
    c. Infra-structure code providing seamless communication between all system components, as well as to a Windows™ based graphical user interface.
3. Collect performance data and statistics from the implemented system (e.g. code and data size, energy consumption, ease of programmability, …) to provide direction for future work on the I-TRM.

This research is work in progress and in future papers we will describe the API definition and implementation of other I-TRM faces. Also experimental results of the overall system implementation are part of future plans for this work.

III. API Definition

Figure 3 below illustrates the six layers of the information face of the I-TRM. As with any other layered model, the lower layers act as service providers to upper layers and as we go up in layers the level os generality of functionality increases. That is the lowest layer (physical layer) deals with most concrete physical level of data acquisition, where as layers above it deal with increasingly more complex issues in transforming data into information. In this section we will describe the function of each layer and the API associate with it.

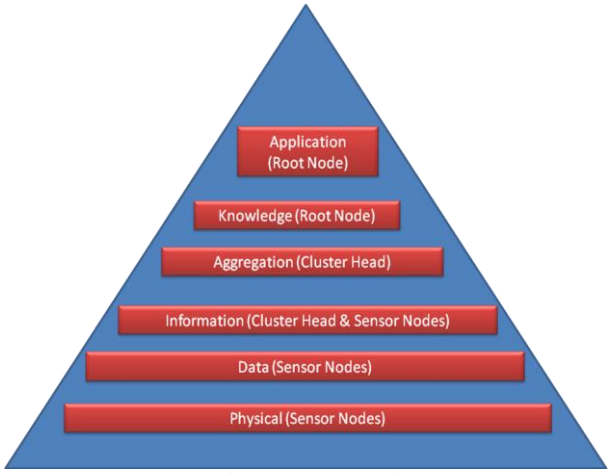

Figure 3 - Information Plane of the I-TRM

A. *Physical Layer:*

This layer constitutes sensors and mechanical units. It gathers raw data in unformatted, unverified and transitory format. It deals with the electrical, mechanical and procedural characteristics. Metadata associated with the physical layer would be the sensor type, serial number, location, and calibration status. This meta-data would generally exist in a stable form as part of the physical sensor.

Implementation of this layer would include the metadata included in Table 1. The API for this layer would include functions such as DataReady(), in the same split phase form as in TinyOS. That is, whenever a data sample is available, the event DataReady() will be created, signaling availability of an A/D data sample. Furthermore specific sensors would have been known at this point to the I-TRM and therefore enumerated and identifiable. There would be also methods available to provide the metadata alone, data with all or a select subset of metadata and data alone. These events will be generated in response to a family of API from the C-TRM having the general form of GetData().

Metadata	Description
Type	Sensor type (e.g. temperature)
Manufacturer	Sensor manufacturer
Model	Sensor model name
Sample size	The size of the generated sample
Sample type	The type of sample (e.g. integer)
AD resolution	A/D resolution (Number of bits)
Sample rate	The sample rate (per second)
Sample rate divider	1 if per second, other int (10, 100…) for slower rates
Location	Location of sensor
Calibration Status	Calibrated or not
Last Calibration Date	Numeric form of "yyyy-mm-dd hh:mm:ss"

Table 1 - IC-TRM Layer 1 Metadata

B. Data Layer:

This layer extracts and transforms data into digital forms and checks the authenticity of the measurements. The voltage from the physical layer is transformed into a byte or a word using a proscribed (although possibly variable) process involving amplifiers, filters and analog to digital converters. Variable parameters could include sampling rate, digitization accuracy, filter cutoff frequency, amplifier gain, etc. Meta-data generated at this level could include these parameters, plus a time tag, a verification bit to indicate that the sensor is calibrated and operating properly, etc. Meta-data from the physical layer and data layer would be bundled with the data to form an informative data packet.

API of this layer would include the capability to have the metadata included in

Table 2 be partially or fully sent with the sensor data.

Metadata	Description
Measurement_ID	Unique identified for this measurement group (e.g. temperature, humidity, pressure1, pressure2, …)
Time Tag	Time tag of sample taken: Numerical form of "yyyy-mm-dd hh:mm:ss:zzz"
Filter Cutoff Frequency	Where applicable, the cut off frequency of the low pass filter
Amplifier Gain	Where applicable, the amplifier gain of the amplifier after the ADC

Table 2 - IC-TRM Layer 2 Metadata

C. Information Layer:

The third layer correlates data with scaling, location, type of measurement, etc, to produce information about the system or environment. the data and metadata from the data layer would be combined to produce information that reports, for example, the temperature at the 12 O'clock position in the combustion chamber of the number one engine was 1000°F at T+1.0 seconds from test start, and that this measurement should be believed with a high degree of confidence.

This layer will have two modes: Transparent Mode and Report Mode. In Transparent Mode, it passes the layer 2 data (and possibly metadata upon request) directly up to layer 4. On the other hand, in Report Mode, this layer uses the data and metadata from layer 2 to create an xml format report. The text string associated with this report can be used directly to generate a sequence of data reports used by applications.

In the Report Mode, the data reported from this layer would have at the minimum, the information listed in

Table 3.

Information	Description
Measurement ID	Unique identified for this measurement group (e.g. temperature, humidity, pressure1, pressure2, …)
Sensor Data	Actual sensor data obtained from layer 2
Layer 1 Metadata	Optional Field
Layer 2 Metadata	Optional Field
Confidence Level	enum (High, Med, Low) This is obtained by a sliding scale of date of last calibration as well as other environmental factors that may affect performance of the sensor. Details of decision thresholds are implementation specific.

Table 3 - IC-TRM Layer 3 Reported Sensor Data

D. Aggregation Layer:

The fourth layer is focused on goal-directed merging of information from various sources, as directed by the requirements of the system or subsystem. For example, readings from multiple temperature sensors, with synchronized time-tags, could be combined to give an instantaneous view of the temperature gradients within the combustion chamber. Additionally, a moving window of a time-sequenced series of readings could be combined to provide the dynamic response to changes in the system. Temperatures, pressures and fuel flows could be combined to create a measure of engine efficiency.

The API for this layer is a set of reported outcomes, based on the particular data fusion, estimation or aggregation method specified in the C-TRM API to this layer. For example if the C-TRM (control face of the I-TRM) layer had requested a data aggregation by taking the moving average of the last N samples and reporting only the average, this API would report the data and the metadata which precisely identifies the meaning and method of derivation of the reported data. Table 4 lists the metadata associated with the data reported from this layer.

Metadata	Description
Measurement_ID	Unique identified for this measurement group (e.g. temperature, humidity, pressure1, pressure2, …)
General Method	General approach taken to reduce the data. This is from an enumerated list.
Specific Method	The specific method of data reduction employed. For example, for aggregation we can have average, min, max, …
Specific Parameters	This field identifies the parameters and constraints of each specific method used in data reduction in this layer

Table 4 - IC-TRM Layer 4 Metadata

E. Knowledge Layer:

This later transforms aggregated information into knowledge by processing against intrinsic and extrinsic information and knowledge available. If the engine temperature approached or exceeded this value, warnings could be issued, or commands could be issued to lower layers in the T&E system to increase sampling rate or accuracy of the engine temperature sensors so a more accurate post-test analysis could be conducted. It should be noted that the knowledge extracted from the lower layers in the IC-TRM will be used to issue the commands by the other TRM faces, and NOT the I-TRM.

The knowledge extraction can be in the form of any of the following rules. Additional rules can be added to this layer per specific application implementation.

- Average value for a subset of sensors has exceeded a certain threshold (re-active)
- The variance (or standard deviation) for a subset of sensors has exceeded a certain threshold – indicating an unstable sensor or sensors (re-active)
- The trend in the last N samples is upwards/downwards, towards an alarming threshold (pro-active)
- Data collected indicates detection of start of an "activity" and thereby requiring change in measurement parameters or engaging additional sensors/mechanisms (pro-active)

The API will comprise of a set of enabling conditions (as listed above and an event report to the B-TRM indicating the specified condition has been met.

Rule List	Rule Types
IC_L5_Event_Report	enum { Average_Exceeded, StdDev_Exceeded, Trend_Alarm, Activity_Start_Detected, Other } ICTRM_L5_Rule_List_t;

Table 5 - IC-TRM Layer 5 Rules

API	Argument	C Data Type
IC_L5_Rule_Specify (Specific Rule for a measurement type being monitored)	Measurement ID	int
	Rule ID	int
	Rule	See **Error! Reference source not found.**
	Rule Arguments	struct { int threshold; } Average_Exceeded_Params_ICTRM_L5_t;
		struct { int threshold; } StdDev_Exceeded_Params_ICTRM_L5_t;

		struct
		{
		int threshold;
		int trend; //upward or downward
		int window_size; //num of samples back
		int percent_proximity; //how close
		} Trend_Alarm_Params_ICTRM_L5_t;

		struct
		{
		// an activity is simply a combination
		// of individual rules
		int rule_ID_1;
		int rule_ID_2;
		int rule_ID_3;
		int rule_ID_4;
		int rule_ID_5;
		int *additional_rules;
		}
		Activity_Start_Detected_Params_ICTRM_L5_t;

Table 6 - IC-TRM Layer 5 API Arguments

F. Application/Presentation Layer:

The uppermost layer provides a means for the user to access and use information from the system in a consistent format.

All event reports of layer 5 are made available to the applications via this layer. This layer will provide a universal and standard interface to all applications utilizing the I-TRM. This interface is based on XML. Extensible Markup Language (XML) is a way to apply structure to a web page. XML provides a standard open format and mechanisms for structuring a document so that it can be exchanged and manipulated. XML offers a standard open format for communities to develop structured mechanisms for marking text and data to facilitate the exchange and manipulation of documents among the community members.

A markup language is the set of rules. It declares what constitutes markup in a document, and defines exactly what the markup means. It also provides a description of document layout and logical structure.

IV. IMPLEMENTATION

A. Software Organization

The implementation of this platform is based on a three tiered software architecture:

1. The sensor mote software: This tier of the software resides in the sensor motes. As can be seen in
2. Figure 4, this tier in itself is composed of two sub-tiers:

   2.1. Layers 1 and 2 that reside in the sensor motes. These motes are the components responsible for ALL data collection in the WSN.

   2.2. Layers 3 and 4 that reside in Cluster Heads. Cluster Heads are essentially sensor motes with perhaps additional computing and storage resources. In a test environment, Cluster Heads can be nothing more than a sensor mote attached to a portable/embedded PC.

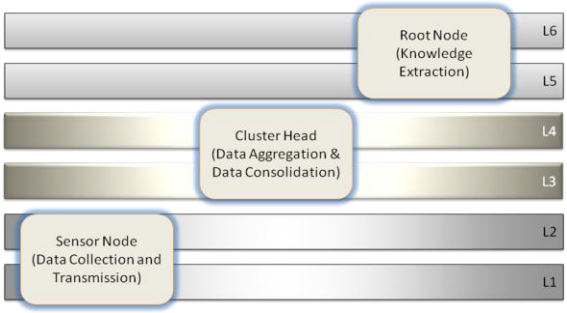

Figure 4 - Location of Software Layers in the WSN Components

3. The user interface: This tier runs as a Windows™ or Linux based application that through its graphical user interface provides the end user visibility into the WSN information collection, and control of the WSN's behavior. Figure 5 illustrates a prototype user interface written in Microsoft Visual Basic™ Express 2008. This tier communicates to the Cluster Heads through the USB interface of the PC on one side, and to the backend server tier through a TCP/IP socket interface on the other. This tier does not contain any decision making intelligence. It acts as a conduit and data format converter, gluing the field deployed portions of the WSN to the user as well as the analysis engine.

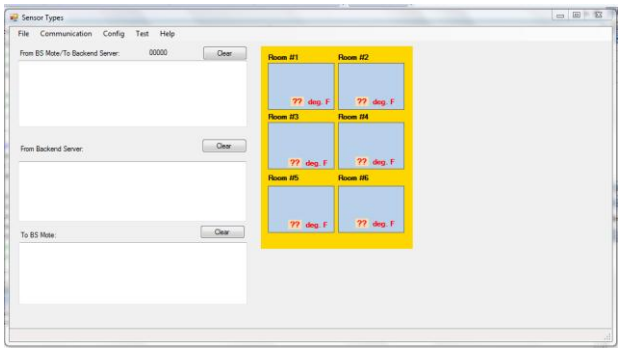

Figure 5 - User Interface Screen Capture

4. The backend server: This program contains the layers 5 and 6 of all faces of the I-TRM (Control, Information and Behavioral). It communicates to the user interface tier through a socket interface. This tier therefore can run remotely from the WSN deployment field, away from potential physical hazards.

The infrastructure software for the sensor motes are as follows:

Operating System: TinyOS is an open source, BSD-licensed operating system designed for low-power wireless devices, such as those used in sensor networks, ubiquitous computing, personal area networks, smart buildings, and smart meters. A worldwide community from academia and industry use, develop, and support the operating system as well as its associated tools, averaging 35,000 downloads a year.

Routing and networking: XMesh is a full featured multi-hop, ad-hoc, mesh networking protocol developed by Crossbow for wireless networks. An XMesh network consists of nodes (Motes) that wirelessly communicate and are capable of hopping radio messages to a base station where they are in range. The hopping effectively extends radio communication

and reduces the power required to transmit messages.

B. Hardware Organization

As can be seen in Figure 6, the test platform consists of N sensor motes that constitute the WSN. A mote, programmed as the base station, collects all upstream packets and forwards them to the user interface in the laptop. In turn, all downstream packets originating from the backend server, are routed through the user interface to the base station and to the intended sensor nodes.

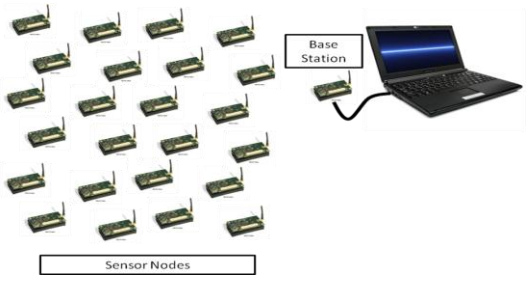

Figure 6 - I-TRM Test Platform Hardware Organization

V. CONCLUSION

As has been shown partially in this paper the I-TRM API is platform independent as well as application agnostic. It can easily adapt to any application by customizing the data structure containing the application specific parameters passing its address to the pointer in the API calls. Very much like the pthreads and other standardized API, the inner workings of the API are abstracted away from the callers, with one major difference that here the inner workings are NOT implemented

only once, but are developed for each custom application. The positive and negative impacts of this API on a system performance are for future study once a full implementation of the system is available. This is mainly because I_TRM does not operate at physical and data link layers. Consequently simulation tools cannot model the complex system level effects of the protocol.

VI. Future Work

Future work in this research is development of an API for the behavior and control faces of the technical reference model (namely B-TRM and C-TRM). Based on specifications of all three faces of the I_TRM, we plan to complete the design and coding a fully functional WSN based on this model.

VII. References

[1] Fortier, P., & Michel, H. (2005). Comparison of the EI TRM versus TENA. ITEA Technology Review Workshop. Atlanta, GA.

[2] Dipple, H., & Michel, H. (2006). The Control Technical Reference Manual. International Conference on Artificial Intelligence. Las Vegas.

[3] Joshi, H., & Michel, H. (2007). Integrating Information-Centric, Control-Centric and Behavior-Centric Technical Reference Models for Autonomous Sensor Networks. Proceedings of the 2007 International Conference on Wireless Networks ICWN, (pp. 319-324). Las Vegas, NV.

[4] Joshi, H. (2008). Autonomous Mobile Sensor Networks Architecture for Hazard Detection and Surveillance. Dartmouth, MA: M.S.,University of Massachusetts Dartmouth.

[5] Michel, H., & Joshi, H. (2008). A Sensor Network Architecture: Information, Control and Behavior Definitions for Large-Scale or Systems-of-Systems Testing. Journal of the International Test and Evaluation Association , 29 (4).

[6] Joshi, H., & Michel, H. (2008). Integrated Technical Reference Model and Sensor Network Architecture. International Conference on Wireless Networks. Las Vegas, NV.

[7] Beheshti, B., & Michel, H. (2011). Middleware/API and Data Fusion in Wireless Sensor Networks. IEEE Long Island Systems, Applications and Technology Conference. Farmingdale.

Biographies

Babak is a faculty member in the School of Engineering and Computing Sciences, New York Institute of Technology, Old Westbury, New York, where he has served since 1987.

Babak received his BEEE and MSEE both in Electrical Engineering from State University of New York at Stony Brook, in 1985 and 1987, respectively. Babak is author of numerous articles and papers, and has presented in many conferences on topics ranging from engineering education to wireless systems. Babak is a recipient of the IEEE Millennium Medal, two IEEE Region 1 awards, two IEEE Long island Section awards and the Ellis College Excellence in Teaching Award. Babak is a member of Tau Beta Pi engineering honor society, and Eta Kappa Nu electrical engineering honor society.

Dr. Howard E. Michel is associate professor of electrical and computer engineering at the University of Massachusetts Dartmouth.

He retired from the U.S. Air Force in 1994, having served as a pilot, satellite launch director, engineer and engineering manager. Other achievements include successfully launching seven satellites by directing launch-base test and integration involving booster, satellite, and range hardware; and developing Department of Defense engineering processes for mission-critical computer systems. Michel is currently a consultant for the U.S. Navy in the area of embedded instrumentation and architectures.