FieldMAP: A Spatio-Temporal Field Monitoring Application Prototyping Framework

James Brusey, Elena Gaura, Daniel Goldsmith, and James Shuttleworth

***Abstract*—The fundamental aim of monitoring is to identify abnormalities in the observed phenomena and allow inference of the likely cause. Faced with the common problems of spatially irregular sensor distribution and intermittent sensor measurement availability, key to fulfilling the monitoring aim is filling in the spatio-temporal gaps in the data. Whilst Wireless Sensor Networks (WSN) technology, combined with MEMS availability potentially offer sensing solutions for a variety of application domains, in the context of monitoring applications a conceptual shift is needed from currently available, point-measurement based "sense-and-send" systems towards the provision of phenomena field representations, in real-time, enabling effective visualisation of the spatio-temporal patterns.**

This paper argues the case for a generic, rapid prototyping framework for end-to-end sensing systems that support the approach of providing field representations for visualisation. A formal approach to framework development was taken, ensuring that resulting instrumentation systems are well specified. Both the framework development and its evaluation are linked to the full cycle of requirements setting, design, and deployment of a prototype instrumentation system for aerospace applications—specifically, health monitoring of a gas turbine engine. The FieldMAP (Field Monitoring Application Prototyping) framework supports multi-modal sensing, provides a number of opportunities for data processing and information extraction, caters for monitoring of the instrumentation health, offers a modular field-mapping design component and allows for real-time phenomena visualisation, data and information logging and post-analysis. Experience with the FieldMAP has shown that sophisticated and robust prototypes can be developed in a short period of time.

I. Introduction

The recent crash of Spainair flight JK 5022 was one of Europe's worst air disasters in decades. Although official investigation is yet to conclude at the time of writing, it is known that the flight had been delayed by high temperatures being shown on an air intake heating system. Technicians had apparently switched off the temperature sensor in order to allow take-off. A critical issue for the investigation will be to identify if there is an underlying design fault with the Boeing MD-82 aircraft. If there is, perhaps better sensing might have helped in the air incident above. For example, if air intake temperatures really were abnormally high, additional independent sensors verifying this would have meant that the problem could not have been be discounted as being due to sensor failure.

This is just an example of imperative need, in safety critical systems, for robust, multi-point sensing instrumentation, whose health can be verified should apparently erroneous data be delivered.

J. Brusey, E. Gaura, D. Goldsmith, and J. Shuttleworth are with the Cogent Computing Applied Research Centre, Coventry University, Coventry, CV1 5FB, UK. Corresponding author is J. Brusey (j.brusey@coventry.ac.uk).

However, retrofitting a fleet of aircraft, for example, with more sensors able to produce independent data streams may not be easy. Apart from the cost of fitting the sensors, wired sensors need associated cabling for signals and power, which adds weight to the aircraft. Wireless sensors have been suggested for some time as an alternative, particularly to reduce the weight burden associated with cabling and to simplify retrofitting.

Similarly, wireless sensors could potentially bring benefits to a variety of system health monitoring applications, not only aircraft health monitoring, and there is indeed a marked drive towards condition-based maintenance (CBM) apparent in the literature for a variety of industrial domains.

Condition-based maintenance is based on the idea that given detailed sensory information combined with a sophisticated model, a good estimate of the likely state of a structure or component can be made, and from this, better decisions can be taken regarding maintenance. (Of course, this is a tightly coupled model based approach which leaves little scope for genericity.)

However, although seemingly a perfect solution for enabling CBM and general aircraft and engines structural monitoring, wireless technologies have not yet been widely adopted. There are a number of issues preventing adoption, such as the need for proven technology reliability and maturity, safety certification, trust, and the conservative nature of many industries, added to the need for electronics that can withstand harsh deployment environments (high temperatures, shocks, and so forth, in aerospace applications, for example).

Should the above listed hurdles be overcome and retrofitting of wireless sensors become a reality, another set of issues are apparent:

- Sensors tend to be spatially distributed according to an irregular pattern, and thus the resulting set of individual sensor readings may be difficult to interpret.
- Individual sensors may fail, in which case they either do not give a reading, or provide a misleading reading. In the latter case, it may be difficult to infer that the sensor has failed by looking at the sensor readings in isolation.
- Wireless sensors may respond only intermittently due to communication failure, and this exacerbates the difficulty of forming a view of the current state of the monitored system.
- Additional redundant sensors may be included but how can this redundancy be best used to improve the quality of the instrumentation output?

Providing a robust networked infrastructure for sets of deployed wireless sensors and enabling the conceptual shift from *data* to *user relevant information* would, theoretically, resolve all above issues.

Nevertheless, answering the specific question of "how should the wealth of data available be used?" is not a trivial undertaking and, so far, the wireless sensing literature does not offer off-the-shelf recipes. Despite the wealth of research over the past few years in the area of WSN, most of the deployed systems put forward are essentially "sense and send" systems, not catering for provision of informational output. Some efforts have been made, however, to resolve the question above in specific application contexts, but the solutions provided are not easily transferable to new domains or new network infrastructures, not to mention new hardware platforms.

In general, filling the *data* to *user information* conceptual gap translates to catering for: a) converting the raw data provided by networked wireless sensors into useful information thus enabling decision making in real-time (noting here that the real-time element is a requirement of most monitoring applications), and b) presenting this information to users (in our case study here pilots, ground staff and test engineers) in such a way as to best help them gain an understanding of the current state of the system being monitored and isolate and qualify abnormalities.

Application specialists and end users of WSN systems clamour for sophisticated information extraction facilities and convenient user interfaces. However, users are rarely able to fully specify, a priori, precisely what information should be extracted and how it should be presented. Fundamentally, the problem stems from a lack of prior availability, in many domains, of detailed measurement systems such as those enabled by WSN and hence lack of prior thinking about how the application would benefit from additional information. Thus, whilst designing a WSN system, it may be a mistake to prescribe and system-encapsulate the inference and usage of the information too tightly, as this would lead to instrumentation that would allow for no discovery in the mining of the data. Another important point here is that the key design requirement for the WSN systems is to evolve as the user knowledge of the monitored phenomena grows.

It is also the author's experience that when designing for safety critical applications, in order for informational output to be accepted, assumptions implicit in any automated inference should be rigorously informed by prior art, field testing, knowledge of the sensed phenomena's spatio-temporal evolution and the properties of the sensors themselves.

It is the very matter of providing WSN systems developers with formalised support in designing instrumentation which caters for informational output that the paper here is concerned with. The logical deductive chain travelled above is applied below to an engine monitoring case study, which motivates and helps evaluate the generic framework put forward in the paper.

Commonly used thermocouple based sensing systems for gas turbine engines take multiple measurements of the high temperatures involved at different points (see an example thermocouple harness in figure 1). However, due to the need

Figure 1. Gas turbine engine annulus thermocouple harness (picture supplied by Vibro-Meter UK)

to keep weight low, only average temperature over a single heavy duty cable for each half of the harness is transmitted to the engine control unit. Although an average temperature measurement is likely to be more reliable than the type of single sensor measurement that delayed flight JK 5022, much information has been lost in the averaging process. Not only does the averaging approach preclude the determination of a detailed picture of the engine gas temperature, which could indicate potential engine problems, but it also prevents the diagnosis of individual sensor faults, de-calibration, and sensor drift and does not allow for sensed data quality assessment.

On the other hand, the use of wireless instrumentation could substantially increase the complexity of the data that could be sent to the engine control unit and hence enable more sophisticated engine control and monitoring. Further, replacing cables with wireless transmissions will reduce the monitoring system weight and, given the availability of detailed temperature profiles for engine control, lead to improved fuel efficiency, reduced carbon emissions and permit a clearer understanding of engine / aircraft health. Thus, the seemingly conflicting CBM measurement system requirements for aircraft components, of low weight and detailed, high rate, robust, multi-site measurement could potentially be achieved through the use of wireless instrumentation.

The benefits of wireless networked instrumentation, however, have the potential to go beyond the weight reduction and informational gains: a wireless system such as the one developed by the authors and presented here as a case study, could allow for the sensors in the network to communicate their "health metrics" with each other, in turn allowing faults and drift to be identified and possibly corrected for in the control systems. This would give much greater confidence in the accuracy of the measured temperature and could, potentially, allow the engine to run with less safety margin and, therefore, more efficiently (with similar benefits on fuel

consumption and emissions).

However, as with many structural health monitoring applications, significant challenges, both technical and in terms of safety certification, must be overcome before networked wireless thermocouples can be embedded into gas turbine engines. The most obvious challenges are to do with the last steps of resolving the application: the instrumentation deployment. The deployment, in general, must be considered first in the design process as it will limit and inform many of the design choices throughout the instrumentation and development process. With regard to temperature measurement, for example, the temperatures outside the casing of the engine can reach in excess of 250°C, precluding the use of most conventional electronic systems. Moreover, maintaining the integrity of an RF signal transmission in an environment that is largely composed of metal whilst not interfering with (or having interference from) other electronic equipment will present major hurdles. Powering the sensors also presents a significant challenge as sensors will need to last for years rather than months or days.

Should hurdles as the ones above be overcome, the benefits to engine management will be significant and could also pave the way for integrating within the wireless instrumentation other types of engine sensors such as vibration sensors, tip clearance and speed sensors.

Given these challenges, it is clear that the alternative of modern smart wired sensors must also be carefully considered as they would answer problems of RF signal transmission and simplify certification while providing many of the benefits of wireless sensors, other than weight reduction. Nonetheless, the fundamental principle of the framework developed here—that of moving more of the processing to the sensor—still applies.

To summarise, whilst detailed phenomena measurement is the aim of emerging WSNs based monitoring systems, sparse, spatially distributed point measurements are hard to interpret. More often than not, the provision of real-time spatio-temporal patterns within the collected data is of interest to the end user of the monitoring system. Such patterns are formed between multi-point measurements and / or between multi-modal measurements. It follows that realising the potential of WSNs in the context of monitoring applications involves a shift from sparse, distributed point measurements towards continuous field representations of the phenomena observed enabling effective visualisation of patterns.

This paper focuses on the development a formal, generic framework for health monitoring applications which enables both understanding spatio-temporal relationships in phenomena that occur over a two-dimensional plane and the rapid production of end-to-end instrumentation system prototypes. In principle, this conceptual approach might also be applied to monitoring phenomena over a three dimensional space.

The main contributions of this work are in providing a framework for rapid prototyping of WSN monitoring applications, and evaluating the use of this framework with the development of a prototype monitoring system for gas turbine engines. The starting point for the work is to first examine, in section II, application requirements for a case study, which involves monitoring gas temperatures over a cross-section of the exhaust of a gas turbine engine. A generic framework is then derived for a wider class of spatio-temporal health monitoring systems. A formal approach to framework development is taken to ensure that, upon applying the framework to a specific application, the resulting prototype system is precisely specified. In section III, the use of the framework for the development of a commissioned prototype of a gas-turbine wireless temperature monitoring system is presented. A discussion of the characteristics of the resulting prototype system is given in section IV, followed by a brief survey of related work (section V) and concluding comments (section VI).

II. Monitoring application requirements

Whilst it is acknowledged that different spatio-temporal monitoring applications have varying requirements (such as different sensor modalities, different spatial and temporal coverage needs, or different timeliness requirements), the approach taken here is to start the FieldMAP development from a specific, well-defined application and extrapolate to a wider class of applications. One of the key issues, however, driving the FieldMAP framework development goes beyond the case study needs as such and draws on the authors' prior work in WSN: the stride to minimise the data transport element and maximise the information transport in WSN systems. In this respect, a proportion of the data processing will tend to be performed in a decentralised way, making use of computation power at remote nodes, while hopefully making some reduction in the size and number of communication transmissions from that node, thus saving power and, for battery powered nodes, extending node life. This general approach, termed here "in-network processing", takes the work beyond the basic "sense and send" philosophy widely applied to real-world WSNs, which tends to push all processing to a central processing node. Commonly, the argument for performing more processing in-network is based on the relative energy cost of computation versus the energy cost of reliable transmission. Since computation is relatively cheap, it appears to be advantageous to perform some computation in the network prior to transmitting on the basis that this computation reduces the number of bits that need to be transmitted [21]. Thus, the underlying philosophy here is to ensure that the framework developed acknowledges and supports in-network processing where it is needed.

A. Specific requirements for gas turbine engine monitoring

1) Instrument related requirements: The end-user base requirements (as established by the engine manufacturer and industry partners collaborating in this work) for the instrument considered here are as follows: 1) Sense the temperature circumferentially / radially at a number of points in a cross-sectional plane of the jet pipe; 2) Transmit temperature measurements to a base station in a way that is secure from malicious or accidental interference; 3) Provide an interpolated, spatio-temporal field view of temperature over the plane being sensed; 4) Store the logged temperatures for later analysis, towards: engine fault diagnosis, and engine

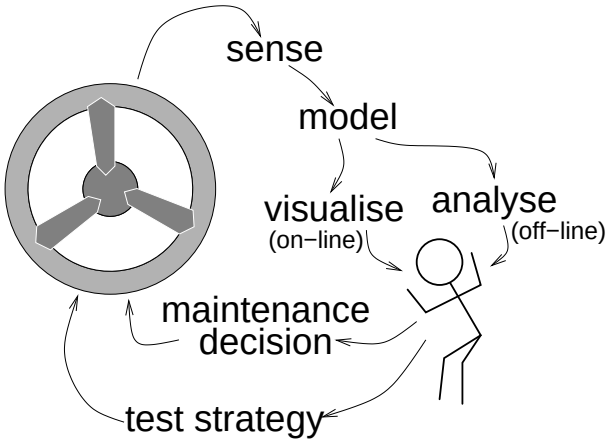

Figure 2. Conceptual flow for the prototype gas turbine engine monitoring system

residual life estimation; 5) Allow for the addition or removal of sensors or nodes; 6) Maintain instrument accuracy and health through integrated calibration and sensor or node fault detection, isolation and management.

Catering for the above requirements means, at a different abstraction level, that one needs to produce a system which:

- seamlessly integrates instrumentation, debugging and performance analysis tools at several development stages,
- allows "plug and play" of several functional components;
- enables organic growth both in terms of the instrument's networking component and in terms of refinement of application requirements;
- enables assessment of how the instrument prototype addresses or should be modified to better address the traditional WSN concerns of overhead, size, and energy.

2) Deployment environment related requirements: Further to the base requirements, there are a number of requirements to do with deploying the monitoring system on a gas turbine engine. Specifically, the system should:

1) Be tolerant to a harsh environment (high temperature, shock, electromagnetic noise);
2) Conform with existing modularity in engine design;
3) Produce minimal electromagnetic interference (EMI);
4) Have minimal size and weight;
5) Have minimal cost at production quantities; and
6) Have low associated deployment and maintenance costs.

At this stage, the focus is on the instrument requirements, rather than the deployment related aspects, such as tolerance of heat and shock. Many of those requirements can be catered for through adequate choices at implementation level, whilst some depend on future advances in related research fields—such as power harvesting for nodes, and miniature, harsh environment electronics.

It is possible to imagine that the conceptual flow for an engine health monitoring system which would encapsulate the requirements above will look like figure 2. Temperature (and possibly other parameters such as vibration in the context here) is sensed at a number of circumferential and radial locations ("sense") within the gas turbine engine. Raw sensor data is noisy and in some cases, sensors may be faulty. Model-based filtering ("model") is used to reduce the effect of sensor noise and thus provide a more accurate estimate of the actual temperature / vibration. Interpolation is also used at this stage to derive a spatio-temporal field function that fits the sensed data. This allows a field representation of the sensed phenomena to be visualised ("visualise") in real-time. Diagnostic features can be extracted from this interpolated field representation either automatically, or by the human expert. Furthermore, offline analysis of sensed measurements over time ("analyse") can be performed.

These two information flows, online ("visualise") and offline ("analyse"), support the human expert in deriving maintenance decisions, either affecting the sensing instrumentation itself (a sensor is faulty and must be replaced) or the engine being monitored. Alternatively, the expert might instead try to get more information by devising a further experiment or test strategy. The flow in figure 2 can be seen as a closed loop system, feeding back changes either to test, to adjust the control logic, or to maintain the engine.

Although not considered here, further control loops such as the above could be designed for the use case where the instrumentation is permanently fitted onto the engine and feeds into the in-flight engine control system. It is indeed here, in the addition of actuation to sensing and the integration of the wireless instrumentation into closed loop systems, that the hope for future applications of the wireless technologies lies. In the authors' opinion, this is also how WSN technologies contribution to increased safety, and active monitoring would be maximised.

B. Generic FieldMAP Framework

Not all wireless monitoring systems will need exactly the set of requirements listed above but most will likely be subscribing to the conceptual flow in figure 2 in part or wholly. This section generalises the requirements in section II-A to a generic framework that will suit many similar health monitoring applications where spatio temporal phenomena are at the heart of the monitoring process. For the purposes of this paper, a spatio-temporal monitoring application is defined as one involving a distributed set of processing nodes $N = \{n_1, n_2, \ldots\}$ where each node n independently performs periodic sensing at a set of locations L_n. All nodes report to a single base station (or sink node), which supports information visualisation and storage. Conceivably, the single sink could be duplicated without a significant change to the architecture.

Each node n is assumed to monitor a set of locations L_n. At any sampling instant t and for each monitored location $\ell \in L_n$, each node must:

1) (**Sense**.) Given a set of sensors $S_\ell = \{s_1, s_2, \ldots\}$ at location ℓ, from the *active* subset $A_{\ell,t} \subseteq S_\ell$, sense a vector of parameters $\mathbf{z}_{\ell,t}$ per location ℓ (which may involve sensed parameters of different modalities, and may consist of a time-series "chunk", such as a 1 second microphone sample).
2) (**Filter and manage faults**.) Transform values from active sensors to an estimate of the state $\mathbf{x}_{\ell,t} \leftarrow$

$f(\mathbf{z}_{\ell,t}, A_{\ell,t}, \mathbf{x}_{\ell,t-1})$. In general, this function must take account of which sensors are active. Also note that the function may make use of the previous state estimate. The resulting state estimate vector can be partitioned into a data portion (corresponding to the state of the location being sensed) and a management portion (corresponding to the state of the sensors). The management portion might include such things as estimates of the reliability or residual life of the sensors. Where there are a series of uncorrelated sensors, the filter may be implemented as a series of functions $f_1, f_2, \ldots$ that each operate on all or part of the state vector $\mathbf{x}_{\ell,t}$.

3) (**Detect events**.) Based on a predicate function $e(\mathbf{x}_{\ell,t}, \mathbf{x}_{\ell,t'}, t')$, where t' is the time of the last transmitted state, decide whether or not to transmit the transformed vector $\mathbf{x}_{\ell,t}$ to the base station (for example, to allow event detection through threshold rules). Note that it is important that the event detection predicate can take into account the last transmitted state as this allows it to identify the value to the receiver of the new state information. Note that here this decision is made locally, without reference to other locations or nodes. The above formulation does not provide for considering global information or even neighbouring locations to aid event detection but this may be an appropriate extension to make in some cases.
   a) (**Queue**.) If an event was detected, append the vector $\mathbf{x}_{\ell,t}$ to the transmission queue.
4) (**Transmit**.) While a channel is available and the transmit queue is non-empty, sort the queue according to some priority ordering $\pi(\mathbf{x}_{\ell,t})$, transmit the first on the queue, and update the last state time t'. The transmission time $t_{transmit}$ and location ℓ are included in the message.
5) (**Schedule**.) Based on the last attempted sampling time for each sensor, and the required sampling rate for each sensor, schedule the next sampling time. In the simplest case, this may involve sleeping for a fixed amount of time, however a discrete event scheduling approach could also be used to allow for different per sensor sample rates are also possible.

The base station has an on-line, real-time component, which performs the following generic procedure:

1) (**Store**.) Store received vectors $\mathbf{x}_{\ell,k}$ for replay, post analysis, and instrumentation health management, along with their associated transmission $t_{transmit}$ and reception $t_{receipt}$ times. Note that the two clocks at the node and base station need not necessarily be synchronised. Without synchronisation, however, it is not possible to determine the latency between transmitting and receiving.
2) (**Update location state**.) Predict the current state as,

$$\mathbf{y}_{\ell,t} \leftarrow \begin{cases} m(\mathbf{x}_{\ell,k}, t-k) & \text{if } k > t^- \\ m(\mathbf{y}_{\ell,t^-}, t-t^-) & \text{otherwise} \end{cases}$$

for all $\ell \in L$, where L is the set of all sensed locations $\bigcup_{n \in N} L_n$, m is a model of the evolution of the state, t^- is the time for which the location state was last updated, and $\mathbf{y}_{\ell,t^-}$ is the previous prediction for location ℓ. The top case is where newer information has arrived, and the bottom where received information is older than that used to estimate the previous prediction. The above approach is required to allow for re-ordering due to priority.

3) (**Identify fresh locations**.) Given a newly received state estimate $\mathbf{x}_{\ell,t}$, the location "age" is updated to $\Delta t_\ell \leftarrow t^* - t$, where t^* is the current time. *Fresh* locations are those whose age is within some limit Δt_{max}; $L_{fresh} = \{\ell : \ell \in L, \Delta t_\ell \leq \Delta t_{max}\}$. Conversely, *stale* locations are those that are not fresh.
4) (**Estimate field**.) Given the current state estimate $\mathbf{y}_\ell = (y_{\ell,1}, y_{\ell,2}, \ldots)^T$, and given a generalised function approximator g, find a parameter vector $\mathbf{w}$ that causes $g_{\mathbf{w}}(p(\ell))$ to approximate the kth dimension of the state $y_{\ell,k}$, for all fresh locations $\ell \in L_{fresh}$, and $p(\ell)$ is the 2D coordinate location of ℓ. In other words, find $\mathbf{w}$ that minimises the sum of errors $\sum_{\ell \in L'} |y_{\ell,k} - g_{\mathbf{w}}(p(\ell))|$. Selection of k is intended to be controlled by the user. Note that the choice of the form of the generalised function approximator $g_{\mathbf{w}}$ (length of parameter vector $\mathbf{w}$ and so forth) will affect how close a fit with observed data will be obtained.
   The *estimate field* step results in a field approximation of the observed phenomena built from the sparse point measurements. Some faults are isolated here due to forming the function approximation solely on data from "fresh" locations.
5) (**Colour map**.) Produce a 2D visual representation based on enumerating a false colour map $c(h_{g_{\mathbf{w}}}(g_{\mathbf{w}}(x,y)))$ for the area being visualised. Note that c is the false colour map, and $h_{g_{\mathbf{w}}}$ is a linear mapping of the range of $g_{\mathbf{w}}$ to $[0,1)$ based on the minimum u and maximum v values for $g_{\mathbf{w}}$ for the area being visualised, $h_{g_{\mathbf{w}}}(k) = (k-u)/(v-u)$ where $v > u$.
6) (**Recall / replay**.) On request, data from a previous period can be recalled and optionally replayed, showing the evolution of the field over a specified period of time.
7) (**Feature / anomaly extraction**.) Identify features or anomalies based on a set of rules, given $g_{\mathbf{w}}$. For example, a typical rule might be to provide an alert if the percentage of the cross-section with a temperature above 900°C exceeds 50%.

Remark 1: Per location ℓ, the node computation is defined by the tuple $\langle A_{\ell,t}, f, e, \pi \rangle$, being the active sensors over time $A_{\ell,t} \subseteq S_\ell$, a filter $f : \Re^a \times 2^a \times \Re^b \rightarrow \Re^b$, event identification $e : \Re^b \times \Re^b \times \Re \rightarrow \{0,1\}$, and a priority ordering $\pi : \Re^b \rightarrow \mathbb{N}$. It is assumed that no two nodes share the same location ℓ, or $L_u \cap L_v = \emptyset$ for all $u, v \in N$. In this definition, $a = |S_\ell|$ is the number of sensors per location, and $b \in \mathbb{N}$ is the dimension of the state vector for a single location. Base station computation is defined by the tuple $\langle N, m, p, g, c \rangle$, being the set of nodes N, a model of the evolution of state at a location m, the mapping of each sensor location to a 2D coordinate position $p : L \rightarrow \Re^2$, a function approximator $g_{\mathbf{w}} : \Re^2 \rightarrow \Re$, and an RGB false colour map $c : [0,1) \rightarrow [0,1)^3$.

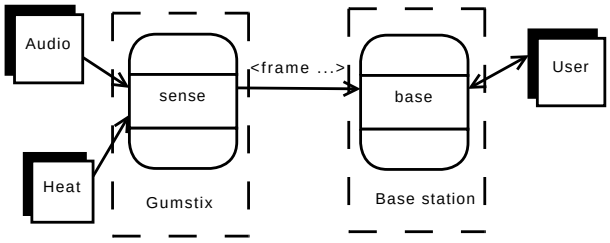

Figure 3. System overview

In order to apply the FieldMAP framework, one must first provide answers to a number of design questions to ensure that the framework best meets the specific application requirements. These questions include:

- How many sensors per location? How many per node? What type of sensors? Sensing wise, is the system mono- or multi-modal?
- What filter f is appropriate? How large a state vector is to be produced? What proportion of this vector caters for sensor health / fault management? What event trigger e to use? What priority mechanism π to use?
- How many nodes? What model m of state evolution to use? What form of function approximation $g_{\mathbf{w}}$ to use?
- How is debugging and problem diagnosis supported? What occurs if a node fails, or a software component is faulty?

The generic framework provided above is explored in section III in terms of its mapping to the specific example of a prototype gas turbine monitoring system.

III. From generic back to specific

To demonstrate how the generic FieldMAP framework can be used, this section attempts to answer the questions above in the context of a commissioned implementation prototype for a wireless gas-turbine monitoring system. Figure 3 shows a high-level view of the implemented system. External phenomena are sensed in two modes: audio and heat. Note that sound sensing acts as a placeholder for vibration sensing, and demonstrates the multi-modal sensing component of the framework. The sensors are attached to wireless processing nodes that communicate with the base-station. The node-level software, referred to here as the SENSE module, is comprised of temperature and sound sensing modules, and a filtering module. The base-station software, referred to here as the BASE module, and which is running on an ordinary desktop computer, is responsible for storing readings in a database, displaying the field visualisation, and interacting with the user.

A. Hardware configuration

For the prototype, pictured in figure 4, two types of sensors were used: microphones and Analog Devices ADT75A IC-based digital thermal sensors. Five locations were sensed per node: four thermal sensors and one microphone. The two sensor types were polled at different frequencies: the microphone at 1Hz and the thermal sensors at 4Hz. Note that this meant that the microphone was considered inactive for 3 sensing cycles out of 4. Also note that IC-based thermal sensors were considered adequate for the prototype but the aim is to eventually support high temperature thermocouple sensors. The sensors were arranged around the outer rim of the jet pipe and also, fixed onto supports, were positioned centrally between the inner and outer rims. As far as possible, the aim was to mimic the positioning of sensors in a live gas turbine engine.

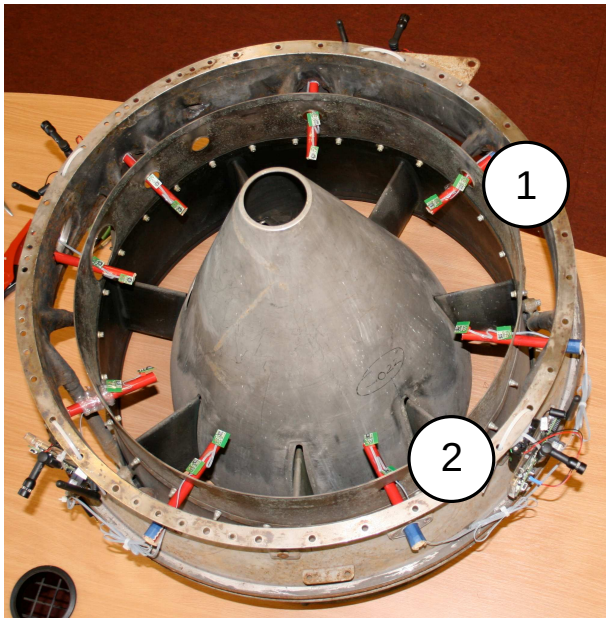

Figure 4. Implemented gas turbine monitoring prototype. Temperature sensors (1) are mounted around the jet pipe between the inner and outer rim and connected via I2C bus to the processing nodes. The nodes (2) are attached to the outside of the pipe. Microphones are mounted directly on the nodes (1 per node).

A key design decision was to identify whether multiple sensors are to be wired to a single node. The deployment environment allows sensors to be connected via wires over short distances. This adds to the wireless system configuration complexity because there are then multiple locations per processing node. However concentrating the processing of a number of sensors in this way makes better use of available processing power and may reduce the cost of the system. For the prototype, four I2C sensors and a single microphone sensor were wired to each node. The I2C bus provides an upper limit of 127 (due to addressing), while the expansion board limited sound processing to a single microphone.

The Gumstix Verdex XM4-BT [1] was selected as the node-level processing and communication platform. The Gumstix Verdex includes a Marvell XScale PXA270 400MHz processor, 16MB of flash memory, 64MB of RAM, a Bluetooth controller and antenna, 60-pin Hirose and 120-pin MOLEX connectors for expansion boards and runs GNU/Linux. The motherboard contains no on-board sensors. Instead, the temperature and audio sensors are connected to the Gumstix board via an expansion board attached to the Hirose connector, that was designed in-house.

B. Node-level processing

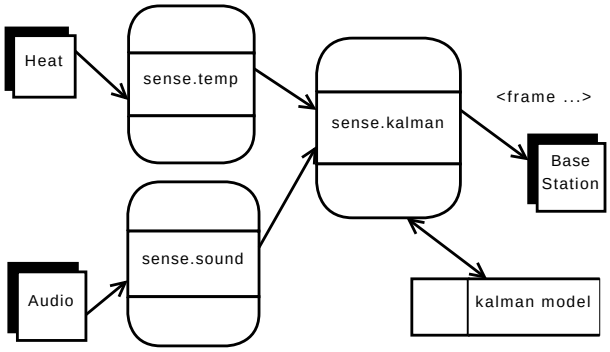

Figure 5. Breakdown of the SENSE component

1) Sense: A breakdown of the SENSE component, which performs all node-level processing, is given in figure 5. The *sense.temp* module queries each temperature sensor in turn via the I2C bus. The *sense.sound* module uses the platform's on-board AC'97 processor and extracts peak sound levels for the last second. The *sense.kalman* module performs filtering on the temperature data, which is described below in more detail. Sound data is not filtered as sound levels were found to be too unpredictable.

2) Filter and manage faults: The generic framework defines a node level filter f. In the gas turbine engine prototype, the filtering stage consists of several processing steps: first, outliers are removed; second, sensor values are calibrated according to sensor-specific calibration coefficients; third a Kalman filter is used to reduce sensor noise.

First, outlier removal is necessary because the sensors occasionally produce extreme values, possibly due to I2C bus communication errors. This is not dealt with by the Kalman filter, which assumes that noise is distributed normally with a zero mean. Second, calibration is required. Although the digital thermal sensors provide degrees Celsius as output and are factory calibrated, sensor readings differed slightly from the temperature measured with a mecury thermometer. Sensor response tends to be quite linear with a roughly unitary slope. Therefore, calibration was restricted here to adding a sensor-specific offset. The calibration coefficient per sensor was obtained by placing sensors in a stirred water bath and comparing with a mercury thermometer.

The third stage of filter processing is to use a Kalman filter [20], [32]. There are two reasons for incorporating Kalman filtering here: a) to recover some resolution in the temperature measurement that is lost through A/D conversion; and, b) to reduce sensor and measurement noise. Temperature in the jet pipe over time is clearly a non-linear function. However, since there are so many factors (both measurable and unmeasurable) affecting it, and since it tends to change relatively slowly, the linear assumption implicit in a Kalman filter is a good compromise. Two possible state models for the filter were considered for this work: one that assumes that the temperature does not change (and thus any change is noise), and one that assumes that the rate of change of temperature is constant (and thus any change in the rate of change is noise). In the present system, the latter was used, as the former tended to introduce some lag when temperature was changing.

Given a single location temperature τ_ℓ, a constant temperature rate model is comprised of the state space for the location $\mathbf{x}_\ell = (\tau_\ell, \dot{\tau}_\ell)^T$, a transition model $\mathbf{F} = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix}$, and some model noise $\mathbf{w}$, such that at time k,

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{w}_k$$

The vector of sensor measurements for a probe $\mathbf{z}_k$ is given by

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k$$

where $\mathbf{H} = (1, 0)$ is the sensor model (corresponding to a single, calibrated temperature sensor) and $\mathbf{v}$ is noise affecting the sensor. Sensor measurement noise is distributed normally with a zero mean and covariance $\mathbf{R}$. It is assumed that sensor noise is uniform across sensors and uncorrelated between sensors and thus $\mathbf{R} = \sigma_z^2$, where σ_z^2 is the variance due to measurement noise. This variance was estimated for the sensors used by measuring a series of values for two sensors at room temperature and taking the individual variance as half the variance of the difference. This estimate is valid if the sensor noise for the two sensors can be considered to be uncorrelated and that the magnitude of the variance is uniform across all sensors. (The sum of variances for uncorrelated random variables is the variance of the sum.) The model noise $\mathbf{w}$ is also distributed normally with a zero mean and covariance $\mathbf{Q}$. In terms of the model, the noise $\mathbf{w}_k$ corresponds to the change in temperature and temperature rate due to the "acceleration" rate of temperature a_k, which is

$$\mathbf{w}_k = \mathbf{G}a_k$$

where $\mathbf{G} = (\Delta t/2, \Delta t)^T$. It is thus possible to derive an expression for the covariance $\mathbf{Q}$ in terms of Δt and the variance in the acceleration rate σ_a^2. The acceleration rate variance provides a convenient tuning parameter to allow for more or less rapid variations in temperature.

A key challenge is to support the Kalman filter with minimal computational cost. In the prototype system, Python was used with matrix manipulation done via Numeric, and although this is a relatively inefficient approach, it reduced coding time dramatically and was still able to run in a real-time mode on the platform of choice for this prototype.

The result of filtering is the production of a state / management vector of the form $\mathbf{x}_{\ell,t} = (\tau, \dot{\tau}, \nu, p_{1,1}, p_{1,2}, p_{2,1}, p_{2,2})^T$ where τ is the temperature, $\dot{\tau}$ is the temperature rate of change, ν is the sound level, and p is the 2×2 estimate covariance matrix. Note that the sound level ν is not processed by the Kalman filter. Also, in the implemented system, sensor locations for sound and temperature were always distinct and thus the state vector contained one or the other but not both.

3) Detect events: No event triggers were used in the gas turbine engine prototype, nor any priority ordering. Event detection could be used to substantially reduce traffic when the system is relatively stable. For example, an event might be "temperature has increased by 2°C." Note that event detection can make use of knowledge of the state evolution model m, and only transmit when the model error would be too large. Under this view, an event might be "based on the last transmitted state, the base station estimate error will exceed

0.5°C." In a sense, the event predicate must make a judgement about the information value of the state vector to the user.

4) Transmit: Priority ordering can be used to deal with environments that only allow intermittent communications. When the channel is lost for a period of time, a FIFO (first-in, first-out) approach to transmitting state is undesirable because only the most recent information is relevant to the display of the current state. As with the event predicate, the question addressed by the priority ordering is "what is the information value of this state vector?"

Note that reordering packets has some side effects that must be dealt with. For example, it is necessary for the event function to consider not the last packet that it processed but rather the last packet that was actually transmitted. Similarly, on receipt, the base station must discard older packets when estimating the current state.

C. Communications

Processing within both the node and the base station assumes the availability of an accurate clock. A contentious issue is whether to require the clocks to be synchronised. In the gas turbine monitoring prototype, NTP (Network Time Protocol) was used to synchronise clocks. However, this produced a dependency that sometimes prevented the system from functioning; if NTP failed to work properly, state data from nodes with old clock values were considered stale and thus discarded. Nevertheless, synchronising clocks allowed the node processing and communication time to be estimated and this was occasionally valuable in identifying nodes that were overloaded or having trouble with transmitting data. Care is still needed to ensure that time synchronisation does not negatively affect the validity of time-based filtering operations.

The Gumstix Verdex motherboard includes a class 2 Bluetooth radio transmitter and antenna. Transmission range is expected to be about 10 metres however actual range varies considerably depending on conditions. Bluetooth was mainly chosen for convenience and to allow the prototype to be rapidly deployed. Bluetooth Network Encapsulation Protocol (BNEP) [13] allows Ethernet protocols to be used. BNEP makes software management, debugging, and problem diagnosis considerably easier as remote shell tools (SSH, SCP) can be used to update software on-the-fly, and interrogate state. Also it allows NTP to be used directly, rather than having to implement a clock synchronisation protocol (such as the Flooding Time Synchronisation Protocol [19]).

In the generic FieldMAP framework, state estimates are sent on a per location basis. In the implemented system, however, different locations that are sampled in the same instant are grouped together in a single, XML-formatted packet and transmitted via UDP.

D. Base-station processing

A breakdown of the BASE component is shown in figure 6.

1) Store: The *base.gather* module is responsible for receiving frames, breaking them up into individual samples, and annotating them with the (x, y) co-ordinate position of their associated sensor. The *base.gather* module also logs all received sensor samples to a remote MySQL database.

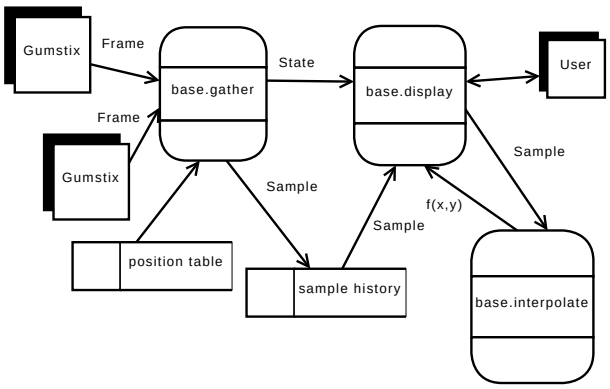

Figure 6. Breakdown of the BASE component

2) Update location state: On receipt of a sample, the *base.gather* module updates its estimate of the current state and reports this to *base.display*.

3) Identify fresh locations: The timestamps of original transmission are used to determine if the value is *fresh* or *stale* (hence identifying a faulty sensor or group of sensors / node). The binary nature of the fresh / stale approach sometimes causes a global change of the interpolated field at the moment that a location undergoes a transition from fresh to stale or from stale to fresh. The change is generally small, unless the state transition coincides with a widespread shift of sensed values, and could be removed by replacing the discrete flag system with a continuous estimate of confidence based on age.

4) Estimate field: In *base.interpolate*, two interpolation methods were implemented, as described below. In order to estimate a field, it is necessary to select a form for the function approximator $g_{\mathbf{w}}$. Two options were explored: a) an inverse-distance weighting (IDW) interpolation originally suggested by Shepard [26]; and, b) a quadratic Shepard interpolation (QSHEP2D) [22]. Gilgen [4] gives a thorough introduction to a variety of other interpolation approaches.

a) Shepard [26] described a basic 2D interpolation function, referred to as an inverse distance weighting (IDW), defined as,

$$f(\mathbf{p}) = \begin{cases} \sum_{i=1}^{N} d_i^{-u} z_i / \sum_{i=1}^{N} d_i^{-u} & \text{if } d_i \neq 0, \forall i \in [1, N] \\ z_i & \text{otherwise} \end{cases}$$

where $\mathbf{p}$ is a vector location in the 2D plane, d_i is the distance from $\mathbf{p}$ to the ith of N sensor locations and z_i is ith sensor value. The exponent u is used to control the smoothness of the interpolation. High values lead to sharp edges between regions while low values lead to soft edges. In this work, an exponent value of 3.2 was used. Inverse distance weighting works well for a small number of data points.

Within the IDW approach, a possible modification can be made to the algorithm to use a non-Euclidean distance measure. In this case, a conical topology may be appropriate as two points on opposite sides of the inner rim are likely to have less correlation than points the same distance apart along the outer rim.

b) A fast version of Modified Quadratic Shepard interpolation, called QSHEP2D, has been developed by Renka [22], [23]. This algorithm is based on fitting a series of quadratic surfaces to the data. In comparison with IDW, QSHEP2D

has the advantage that it does not assume a zero gradient at sensor locations, it has a lower computational cost for a large number of data points, and considers both the direction of an influencing data point, not just the magnitude [26].

While these are general purpose interpolation algorithms, they are good examples of simple interpolation (IDW) and surface fitting (QSHEP2D) algorithms. The quality and suitability of the estimated field could be improved for a specific application by using more sophisticated, physics-based models of the propagation and diffusion of the relevant phenomenon.

The limiting factor for display frame rate was time to calculate the interpolated surface $g_{\mathbf{w}}(x, y)$ for all points x, y being displayed. As a compromise, the display was divided into blocks with a block size that is adjusted automatically to keep the frame rate at a reasonable level. The interpolation function was only calculated once per block.

As described in the *estimate field* step of the generic base-station procedure, the user can choose which aspect of the state vector to produce a field map for. For the gas turbine prototype, the user was able to switch between temperature, temperature rate, and sound levels.

5) Colour map: A standard thermographic false colour map was used. False colour mapping was dynamically scaled according to the minimum and maximum values of the interpolated surface within the bounds displayed. In the implementation, it was found useful to place a ceiling on the scaling factor used for the case where the minimum and maximum field values are close together. In terms of displaying temperature, this means that when all temperatures are approximately the same, the surface shows a single value, which for the false colour map used, was displayed as all black.

6) Recall / replay: Recall and replay were considered of lesser priority in the prototype system but are still supported by querying the database. During the prototyping phase, the emphasis for recall and replay tends to be on ensuring that the system is functioning correctly, analysing sensor characteristics and so forth. It is expected, however, that the ability to replay past events will be critical to understanding, in detail, phenomena that have occurred during a flight or ground-test, and will thus enable the development of expert system rules for anomaly detection.

7) Feature / anomaly extraction: Sensor failure / anomaly detection has been implemented in the form of a leave-one-out cross validation of the interpolation. For each sensor location, an interpolated estimate for that sensor location is formed based on all other sensors and excluding the sensors at that location. The MSE (mean squared error) derived from this provides an indication of appropriateness of the interpolation method. Assuming that the interpolation is appropriate, a large (relative) error for a particular sensor location provides evidence indicating a sensor fault.

The above method is similar to an approach suggested by Kobayashi and Simon [15], which uses a bank of Kalman filters to detect sensor faults in aircraft engines. In their scheme, the ith filter makes the hypothesis that the ith sensor is faulty and should be excluded. When a sensor fails, all filters will show large fault indicator signals except for the filter with the correct hypothesis.

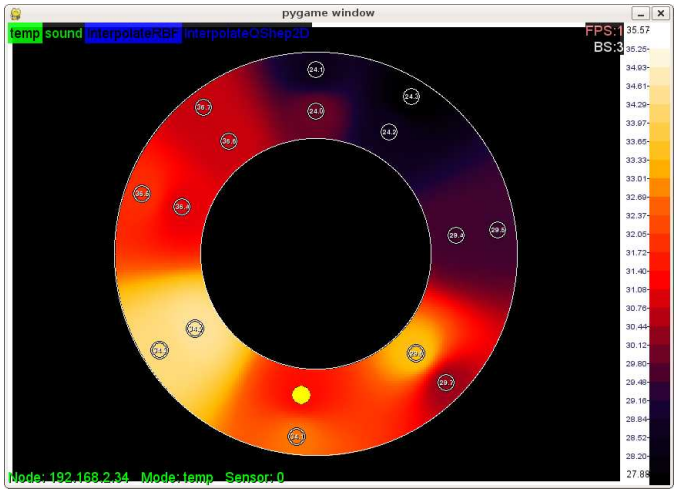

Figure 7. Temperature field map produced at base-station user interface. Small circles represent temperature sensors. The filled circle is the one currently selected for examining history (not shown here).

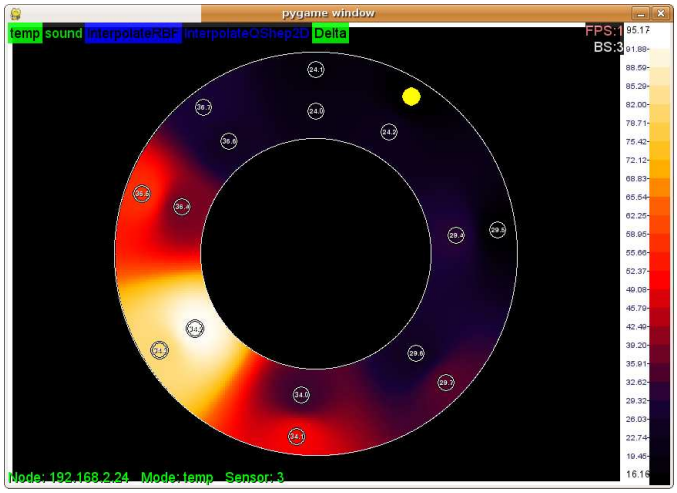

Figure 8. Temperature rate of change field map corresponding to the same time as figure 7. Units for the legend are in $0.001°\text{C}.s^{-1}$.

It is planned to identify contour lines (lines of equal value or isopleths), and to extract area sizes of different contours. This will feed into expert system rules to identify anomalies. Previous work [27] has explored the issue of identifying contours for WSN-based data in more detail.

IV. Discussion

Sample visualisation results for the case study treated here, are shown in figures 7 and 8. The images were captured during experimentation involving heating one section of the prototype pictured in figure 4. Figure 7 shows the real-time, online generated temperature field map at a time instant, which has maximum values of around 36°C at two hot spots (lower left and lower right) and a minimum value of around 28°C at a cooler region towards the top right. Interpolation between measured positions was performed using Shepard's Inverse Distance Weighting (IDW). Figure 8 shows rate of change of temperature $\dot{\tau}$ for the same time sample as figure 7. A maximum temperature increase of about $0.1°\text{C}.s^{-1}$ is focused

Table I
COMMUNICATIONS AND PROCESSING LATENCY

	Latency (seconds)		
	Min	Median	Max
Transmission without filtering	0.01	0.09	5.10
Transmission plus filtering	0.03	0.28	8.48
with separate net	0.09	0.26	15.38
with extra nodes	0.10	0.30	10.10

in the lower left area, while a more gentle temperature increase of $0.02°\mathrm{C}.s^{-1}$ is occurring in the rest of the annulus. The two contrasting field maps correspond to two key questions that the user of a system, such as the one treated here as a case study, might want to ask: what is (relatively) hot, and what is getting (relatively) hotter? The maps thus provide a demonstration of many of the key components of the FieldMAP framework and its informational benefits to engine monitoring.

The prototype system was evaluated in terms of robustness, performance, and accuracy. Robustness-wise, the system was stable over a number of long running (up to several weeks) trials.

Experimental results for the system in terms of communication and processing latency are shown in table I. Packet transmission times were measured by first ensuring that NTP had synchronised the clocks and then starting the system and measuring time between acquiring a temperature sample and reception of the sample at the base station. Higher than average latency was measured during start up, accounting for the large maximum transmission times observed. Without filtering, the median processing and transmission time is around 0.09 seconds. Including the Kalman filter increases this significantly to around 0.28 s. Injecting noise into the network, either by adding a separate Bluetooth network, or by adding additional nodes, both with ping traffic, makes little difference to the median but increases the worst case values significantly. Note that these results were obtained for communication of sensor data formatted as XML, which added significant overhead to the packet size.

Leave-one-out cross validation [16] was used to allow comparison of the different interpolation approaches implemented. Apart from IDW and QSHEP2D, a naive "nearest neighbour" interpolation was also implemented that estimates a point's value as being the same as its nearest neighbour. The experimental set up used in validation is shown in figure 9. Lamps were used to induce temperature changes, while polyurethane film was used to reduce external airflow. Table II shows the results from a 30 minute trial, using 20 temperature sensors sampling at 4Hz and which included a series of heating and cooling cycles. For both temperature and rate of change of temperature, IDW produced the least error. Surprisingly, the nearest neighbour approach produced better results than QSHEP2D. QSHEP2D would be expected to perform well for continuous, smoothly changing phenomena. However it seems likely that convection and shadows meant that both air temperature and rate of change of temperature varied greatly even for nearby points.

In summary, the FieldMAP framework provides a recipe for rapidly building prototype wireless end-to-end monitoring

Figure 9. Experimental configuration including lights to heat pipe section and polyurethane film used to reduce airflow.

Table II
CROSS VALIDATION RESULTS

	Mean squared error (MSE)	
	Temperature $(°\mathrm{C})^2$	Temp. Rate $\left(°\mathrm{C}.s^{-1}\right)^2$
IDW	0.57	2.6×10^{-5}
NN	0.83	4.0×10^{-5}
QSHEP2D	0.99	5.1×10^{-5}

systems. Although gas turbine engine monitoring provides a focal point for discussing the approach, the framework is general. Rapid prototyping is emphasised as this is seen as the best way to reduce the risk of development and to help the end user to formulate requirements as early as possible. In comparison with much of the work on WSN middleware, the framework integrates information extraction, both in terms of node level inference of state from sensor readings all the way to presenting the end user with information in an interpretable form. Not all aspects of the framework will be critical to all applications and the case study demonstrates the plug-in nature of the framework implementation, both in the sense of optionally leaving out some aspects (such as event detection), and in including multiple choices for others (such as multiple interpolation schemes).

In the FieldMAP framework, as presented in this paper, a key simplification has been made: to focus on *local* filtering and event detection. This was despite a philosophical urge to explicitly enable in-network processing. Although attempting to make use of global information when performing filtering or event detection locally is likely to be impractical in a scalable system, neighbouring nodes might be enabled to

communicate their state to one another. Enabling the use of neighbouring state information during filtering and event detection, for example, should be a straightforward extension of the FieldMAP framework. In the case study presented, this might be used to support a predictive model that takes account of heat transfer between neighbourhoods, for example.

V. Related work

Over the past decade, many WSN-based monitoring applications have been proposed. Some of them have been implemented, but only a few have been deployed [2], [3], [5], [8], [7], [14], [28], [30]. Common to all deployments is their specificity to the tackled application and the steep learning curve the developers took towards deployment. It appears from the literature that each and every of the systems cited above was developed and deployed as a one-off system, responding to a specific set of constraints, mainly dictated by the application at hand. For example, a sound-analysis WSN application, such as VoxNet [2], focuses on the timeliness of the data rather than node-size, battery life, or scalability. Here, in network processing is essential to satisfy the timeliness constraints. A body sensor application, such as CodeBlue [18], focuses on miniaturisation and reducing power consumption. These are two examples but many more have been reported in the literature with the common trait of limiting the design space according to the application at hand.

Taking a bottom-up view, extensive WSN work revolves around development of reliable, energy saving communication protocols (e.g. Directed Diffusion [9]), energy autonomy at nodes through self-power (e.g. Perpetuum [10]), and low power embedded operating systems (e.g. TinyOS [6]).

It is hence difficult to evaluate a generic framework such as the one proposed here against past application related development work or achievements in the development of particular WSN components. Nevertheless, the call for a generic framework has been put forth to the WSN research community several times [25], [24], [31]. It is this call that the framework proposed here is attempting to answer.

In recent work by Werner-Allen *et al.* [33], the Lance framework has been developed to respond to the needs of WSN-based high data rate applications (specifically seismic monitoring of volcanos). Lance is informed by past experience with volcano monitoring that demonstrated the need for prioritisation of messages, summarisation of data locally, and energy management. Lance is, by and large, a centralised approach and best suited to the data rate for seismic monitoring. In comparison, FieldMAP takes the approach of estimating state at the node, giving authority and a strong foundation to nodes to determine what to transmit and when. Notably, both frameworks avoid inter-relationships and thus dependencies between individual nodes that might lead to individual failures causing more widespread data loss. Lance focuses on capturing the most useful data for future off-line analysis, whereas FieldMAP supports both off-line and on-line data capture.

More generally, plug-in components with FieldMAP, such as the filtering mechanism or feature extraction are expected to build upon traditional condition monitoring techniques, such as model-based sensor fusion, sensor fault detection, and so forth. Furthermore, the FieldMAP framework is intended to be used in conjunction with modern routing protocols that are intended to extend battery life and cope with intermittent communication links in an ad hoc, multi-hop network. With regard to the specific case study used here for evaluating the framework, there is considerable work in terms of wired monitoring systems for gas turbine engine monitoring [17], [11], [29]. Jaw [11] reviews several open architectures that have been proposed for generic CBM, including MIMOSA's OSA-CBM, and ISO's six-layer CBM model. The OSA-CBM architecture, for example, consists of several layers: Data Acquisition (DA), Data Manipulation (DM), State Detection (SD), Health Assessment (HA), Prognostics Assessment (PA), and Advisory Generation (AG). Although the open architectures above are broadly similar to the FieldMAP framework, there are specific distinguishing features in FieldMAP, introduced or enabled by wireless transmission, such as the opportunity for in-network processing, event detection, and message prioritisation. In terms of similarities, Litt *et al.* [17], for example, point out the need for the inclusion of a high frequency data analysis module in diagnostic architectures (such as a Fast Fourier Transform (FFT)) for dealing with vibration sensors. This corresponds well with the approach of in-network processing taken in FieldMAP, where such analysis can be performed locally, thus reducing the need for high bandwidth transmissions. Hence, overall, the FieldMAP framework conforms well with established requirements for CBM but adds real-time information extraction benefits and the prospect of rapidly deployable extensions to existing architectures to better support heterogenous multi-point sensing and spatio-temporal phenomena characterisation.

Despite existence of standards guiding engine monitoring system (EMS) design for decades, Tumer and Bajwa [29] noted in 1999 that most commercial aircraft engines did not support sophisticated engine monitoring. More recently, calls for further unification of frameworks and standardisation of approaches to enable greater adoption continue [11]. Considering the conservative nature of the aerospace industry and given the added complexity of using wireless sensors rather than wired ones, it may be sometime before a standard wireless monitoring framework for aerospace becomes widely accepted. Nevertheless, fault identification and residual life estimation remain a high priority for the industry [11], [17] and for this application domain, research into self-monitoring, self-healing, self-* approaches is perceived as important.

Lastly, WSN main selling point comes from the promise of enabling real-time building and delivery of continuous, spatio-temporal patterns, equating in essence to production of field maps, such as the one proposed here. Approaches to producing a field map or contour map from sparse, irregularly spaced sensor measurements have long been used in the geosciences domain [4] without reliance on WSN technology. It is only more recently, that field mapping has begun to be explored in the context of sensor networks [12], [34], hence making this work timely with respect to the means of extracting and visualising information about the monitored phenomena.

VI. Conclusions

The contribution of this work is two fold: on the one hand, a novel framework for rapid prototyping of WSN monitoring applications has been developed and evaluated; on the other hand, the case study has produced a prototype monitoring system for gas turbine engines.

The main thrust of the framework was to enable both the understanding of spatio-temporal relationships in a monitored phenomena and the rapid production of appropriate monitoring system instrumentation prototypes. A formal approach was taken to the framework development to ensure that resulting systems are well specified. The framework was evaluated through the above mentioned case study, which successfully produced a working prototype meeting the set requirements. It should be noted that the prototype system does not support the high temperatures, harsh conditions, nor some other specific deployment requirements such as the need to ensure that communications are invulnerable to accidental or malicious interference. Nevertheless, the rapid prototyping approach enabled by the framework has already helped the end user focus and formulate specific information extraction requirements (such as the mapping of rate of change of temperature).

In comparison with other work in the aerospace domain, the FieldMAP framework resulted in a prototype with reduced bandwidth requirements through in-network state estimation and event detection. Due to the focus on safety-critical applications here, interaction between nodes is explicitly avoided as this might lead to unnecessary interdependency and reduce system robustness.

It is expected that the main reasons that WSN application developers will make use of the FieldMAP framework are: the overall simplicity of the approach, leading to lower system complexity and, hopefully, fewer bugs; the incorporation of in-network processing for event detection and state estimation, leading to less communication traffic and thus lower energy requirements and extended battery life; the emphasis on starting with general purpose field-based visualisation of phenomena allowing a gentler development curve and facilitating user-requirements capture.

In future work, it is planned to continue hardware development on the prototype towards a robust, low power, wireless sensing system that can withstand the high temperatures and harsh environment of a gas turbine engine. Also, a generic open-source toolkit and guide supporting FieldMAP-based prototyping is to be developed. It is aimed to expand the framework to cater for more sophisticated neighbourhood based local processing and event detection, for use in applications of a non-critical nature.

Acknowledgements

The project, within which the work described here represents the first step, is funded through the an Engineering and Physical Sciences Research Council grant in collaboration with Vibro Meter UK Ltd and TRW Conekt Ltd (suppliers and developers of engine temperature sensors and harsh environments electronics, respectively). The authors wish to thank the anonomous reviewers for their insightful comments, which have helped improve the paper.

References

[1] Gumstix motherboard io [online] http:/docwiki.gumstix.org/, 11th September 2008. accessed 11th September 2008.

[2] M. Allen, L. Girod, R. Newton, S. Madden, D. T. Blumstein, and D. Estrin. Voxnet: An interactive, rapidly-deployable acoustic monitoring platform. In *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, pages 371–382, 2008.

[3] Kameswari Chebrolu, Bhaskaran Raman, Nilesh Mishra, Phani K. Valiveti, and Raj Kumar. Brimon: a sensor network system for railway bridge monitoring. In *MobiSys '08: Proceeding of the 6th international conference on Mobile systems, applications, and services*, pages 2–14, New York, NY, USA, 2008. ACM.

[4] Hans Gilgen. *Univariate Time Series in Geosciences: Theory and Examples*. Birkhäuser, 2006.

[5] P. Greenwood, D. Hughes, B. Porter, P. Grace, G. Coulson, G. Blair, F. Taiani, F. Pappenberger, P. Smith, and K. Beven. Using a grid-enabled wireless sensor network for flood management. *Demo at Eighth International Conference on Ubiquitous Computing (UBICOMP'06) Demo Session, Orange County, California, USA, Sept*, 2006.

[6] Jason Lester Hill. *System architecture for wireless sensor networks*. PhD thesis, University of California, Berkeley, 2003. Adviser-David E. Culler.

[7] D. Hughes, P. Greenwood, G. Blair, G. Coulson, F. Pappenberger, P. Smith, and K. Beven. An Intelligent and Adaptable Grid-Based Flood Monitoring and Warning System. *Proceedings of the UK eScience All Hands Meeting*, 2006.

[8] Danny Hughes, Phil Greenwood, Geoff Coulson, and Gordon Blair. Gridstix: Supporting flood prediction using embedded hardware and next generation grid middleware. In *WOWMOM '06: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, pages 621–626, Washington, DC, USA, 2006. IEEE Computer Society.

[9] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Mobile Computing and Networking*, pages 56–67, 2000.

[10] E.P. James, M.J. Tudor, S.P. Beeby, N.R. Harris, P. Glynne-Jones, J.N. Ross, and N.M. White. An investigation of self-powered systems for condition monitoring applications. *Sensors and Actuators A*, 110:171–176, 2004.

[11] Link C. Jaw. Recent advancements in aircraft engine health management (EHM) technologies and recommendations for the next step. In *Proc. ASME Turbo Expo 2005*, number GT2005-68625, Reno-Tahoe, Nevada, June 2005.

[12] Guang Jin and Silvia Nittel. Toward spatial window queries over continuous phenomena in sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 19(4):559–571, 2008.

[13] P. Johansson, M. Kazantzidis, R. Kapoor, and M. Gerla. Bluetooth: an enabler for personal area networking. *Network, IEEE*, 15(5):28–37, 2001.

[14] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li S. Peh, and Daniel Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. *SIGOPS Oper. Syst. Rev.*, 36(5):96–107, 2002.

[15] Takahisa Kobayashi and Donald L. Simon. Application of a bank of Kalman filters for aircraft engine fault diagnostics. In *Proc. ASME Turbo Expo 2003*, number GT2003-38550, Atlanta, USA, 2003.

[16] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. pages 1137–1143. Morgan Kaufmann, 1995.

[17] Jonathan S. Litt, Donald L. Simon, Sanjay Garg, Ten-Heui Guo, Caroline Mercer, Richard Millar, Alireza Behbahani, Anupa Bajwa, and Daniel T. Jensen. A survey of intelligent control and health management techologies for aircraft propulsion systems. *J. Aerospace Computing, Information, and Communication*, 1(12):543–563, 2004.

[18] David Malan, Thaddeus Fulford-Jones, Matt Welsh, and Steve Moulton. Codeblue: An ad hoc sensor network infrastructure for emergency medical care. In *International Workshop on Wearable and Implantable Body Sensor Networks*, April 2004.

[19] Miklós Maróti, Branislav Kusy, Gyula Simon, and ákos Lédeczi. The flooding time synchronization protocol. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 39–49, New York, NY, USA, 2004. ACM.

[20] Peter S. Maybeck. *Stochastic models, estimation, and control*, volume 141 of *Mathematics in Science and Engineering*. 1979.

[21] G. J. Pottie and W. J. Kaiser. Wireless integrated network sensors. *Commun. ACM*, 43(5):51–58, 2000.

[22] Robert J. Renka. Algorithm 660: QSHEP2D: Quadratic shepard method for bivariate interpolation of scattered data. *ACM Trans. Math. Softw.*, 14(2):149–150, 1988.

[23] Robert J. Renka. Multivariate interpolation of large sets of scattered data. *ACM Trans. Math. Softw.*, 14(2):139–148, 1988.

[24] Kay Römer. Programming paradigms and middleware for sensor networks. In *GI/ITG Workshop on Sensor Networks*, pages 49–54, 2004.

[25] Kay Römer, Oliver Kasten, and Friedemann Mattern. Middleware challenges for wireless sensor networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(4):59–61, October 2002.

[26] Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517–524, New York, NY, USA, 1968. ACM Press.

[27] J. K. Shuttleworth, E. I. Gaura, and R. M. Newman. Surface reconstruction: Hardware requirements of a som implementation. In *Proceedings of the ACM Workshop on Real-World Wireless Sensor Networks (REALWSN'06)*, pages 95–96, June 2006. ACM ISBN: 1-59593-431-6.

[28] Gilman Tolle, Joseph Polastre, Robert Szewczyk, David Culler, Neil Turner, Kevin Tu, Stephen Burgess, Todd Dawson, Phil Buonadonna, David Gay, and Wei Hong. A macroscope in the redwoods. In *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, pages 51–63, New York, NY, USA, 2005. ACM Press.

[29] Irem Y. Tumer and Anupa Bajwa. A survey of aircraft engine health monitoring systems. In *Proc. 35th Joint Propulsion Conf.* AIAA, 1999.

[30] Ville Tuulos, Jürgen Scheible, and Heli Nyholm. Combining web, mobile phones and public displays in large-scale: Manhattan story mashup. *Pervasive Computing*, pages 37–54, 2007.

[31] H. Vlado, Andreas Köpke, Holger Karl, Adam Wolisz, and Technische U. Berlin. A common wireless sensor network architecture ?

[32] Greg Welch and Gary Bishop. An introduction to the Kalman filter. Technical report, University of North Carolina at Chapel Hill, 1995.

[33] Geoff Werner-Allen, Stephen Dawson-Haggerty, and Matt Welsh. Lance: Optimizing high-resolution signal collection in wireless sensor networks. In *Proc. 6th ACM conference on Embedded Network Sensor Systems (SenSys '08)*, pages 169–182, New York, NY, USA, 2008. ACM.

[34] Wenwei Xue, Qiong Luo, Lei Chen, and Yunhao Liu. Contour map matching for event detection in sensor networks. In *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 145–156, New York, NY, USA, 2006. ACM Press.