SWORDFISH: a Framework to Formally Design WSNs Capturing Events

Simone Campanoni, William Fornaciari
Politecnico di Milano, Dipartimento di Elettronica e Informazione
P.zza L. Da Vinci, 32 – 20133 Milano, Italy
email: {campanoni,fornacia}@elet.polimi.it

***Abstract*: A Wireless Sensor Network (WSN) consists of spatially distributed autonomous devices equipped with sensors and radio communication capabilities. They can act in a cooperative manner for monitoring environmental parameters like pressure, temperature, acceleration, light, humidity, etc.**
The typical WSN design problem is to discover a proper set of sensors and their spatial distribution, so to enable the monitoring of relevant parameters for the application, while concurrently optimizing some design goals (e.g., cost, reliability, lifetime and energy requirements).
In this paper we present part of the SWORDFISH (Sensor netWORks Development Framework Integrating Simulation and Hardware optimization) project, aiming at providing a design and verification environment for WSNs, including in the design loop the simulation of physical events and a formal verification of the WSN desired properties against the network optimization goals.
In particular, it is presented the overall design framework, the approach to model properties and goals of the network and some representative case studies showing the value of a toolsuite gathering WSN formal optimization/planning and environment simulation.

1. INTRODUCTION

WSN are becoming a popular solutions for many application fields, where monitoring of physical parameters is crucial or even the main application goal. Current sensor technologies are shrinking form factors and wired connectivity requirements, dramatically. Nevertheless, many other questions are still open, like optimization of overall costs (sensors, communication infrastructure, networks deployment…), feasibility analysis to understand suitability and effectiveness of the WSN against real application goals, lifetime (especially in the case of battery operating sensor nodes), robustness, etc [1]. The main lack is the missing of an overall analysis and design framework, to enable a quantitative evaluation of the above properties, taking into account not only the networking-related issues or the distributed software system itself, but also the cross relations existing among the network topology, the nodes, the environment where the WSN is embedded and the events to be monitored, namely the real and comprehensive functional goal of the WSN.

During the last decade, in literature appeared a number of proposals regarding simulation and deployment of WSNs; some of the more mature and publicly available results are listed in [2-12]. Each of these proposals addresses with meaningful results some specific simulation or implementation level aspects of WSN analysis, covering hardware, software and networking. Unfortunately, from the best of our knowledge, up to now none is addressing with a proper and formal extent the capability of the network to capture the events to be monitored, since the primary focus is frequently related to the optimization of the cost or to verify other properties like power consumption, robustness of the connection layer or the analysis of the middleware-level models of computation.

The focus of this paper is a wide class of applications where, in addition to the typical monitoring capabilities, it is also required highlighting the occurring of particular events. Under these assumptions, our methodology to tackle the problem of designing a sensor network requires to: specify the characteristics of the events to be discovered; select a proper set and type of sensors to enable the capturing of such events; embed the sensors in the environment in a way to formally ensure the capturing of the desired events while optimize some design goals. Note that this approach is so top-down and general to embrace both wired and wireless sensor networks. Such an approach to the design is the baseline of the SWORDFISH project, whose main features concerning system-level modeling and design of WSNs are here addressed.

The paper is organized as follows, Section 2 sketches the overall architecture of SWORDFISH; Section 3 discusses the models of the events to be recognized and the design flow to create a WSN ensuring that all the events can be sensed. Some simple but representative examples are discussed in Section 4 and Section 5 provides details regarding the software implementation of SWORDFISH. Finally, last Section draws some conclusions and outlines the next steps of our research.

2. SWORDFISH AT A GLANCE

In a nutshell, the SWORDFISH framework has a graphical user interface to describe the actors (sensors, network, events, and environment) and the design goals of the systems (properties of the network and target optimization parameters), whose roles are explained below and depicted in Figure 1.

Environment Editor. This module allows defining a simplified representation of the environment where the WSN will be deployed, with graphical views of the associated physical parameters (e.g., temperature, humidity, 3D-spatial representation, obstacles …) and the possibility to specify constraints such as position and type of some sensors, if relevant for the users.

Event Editor. The purpose of this editor is to support the description of the events to be captured in terms of variation of some physical parameters to be sensed, along with their timing characteristics. The models are flexibly implemented via plugins.

Sensor Editor. It is the mean to obtain the analytic representation of the sensing nodes, which is a modeling of the relation existing between the sensed physical parameters and the signal produced. The

model of the node includes additional information like cost, type of sensors, energy consumption, accuracy, speed, etc.

Network Editor. In addition to the node features, a model of the available connection channels among nodes is specified. In general, this model can cover both wired and wireless links, although in our first implementation we focused on wireless only.

Predicate Editor. This editor allows the user to specify via a logic formula the properties to be verified in the case a given event occurs. This is of paramount importance to verify that a WSN is actually capable to argue if an event is recognized, or, dually, to select the proper set of sensors to recognize the events. This is a concept more abstract and powerful then a simple measurement.

Simulation Kernel. It is the engine which, based on a simulation of the event occurring, modifies the configuration of the world model accordingly. This allows feeding the sensor node models with the real (location aware) data of the world, including their dynamics. Hence, both the physical parameters of the environment and the events to be monitored can be jointly modeled and verified by the *Predicate Analyzer* (see Figure 1).

Optimization Editor. It is an editor allowing the designer to specify and tune the goal functions and the formal model of the network properties/constraints.

Planner. This is the main module for both verification and network design. It allows to formally verifying that a given WSN is able to capture a set of events as well as to support the building and optimization of the overall network according to the selected policies and goals.

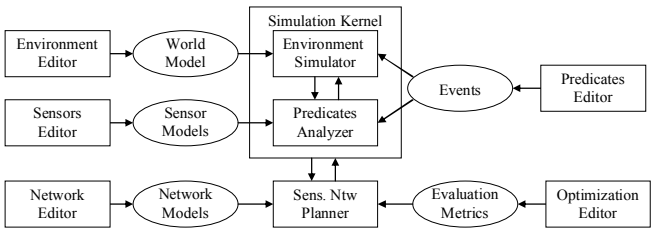

Figure 1. Coarse-grain architecture of SWORDFISH.

The conceived design flows, require the availability of a repository of sensor models for the different type of physical parameters we are interested, together with a description of the connection capabilities and properties. In the first version, the network models have been obtained directly interfacing TOSSIM, but the current implementation is based on the development of a new custom version to rise up the abstraction level of the modeling.

Being the overall software toolsuite organized as a set of plug-ins, possible replacement or improvement of the above models is straightforward. SWORDFISH architecture aims at allowing the users to deal with the following problems:

- *Verification*. Determine the occurrence of a set of events (e.g., fire in a defined region, presence of water, temperature and humidity over a certain threshold for a time window, etc.) for a given sensor network.
- *Design/Planning*. Given a set of events and some constraints/goals, the task is to discover the optimal sensor network capable to identify the events while maximizing a user-controlled goal function.
- *Sensitivity Analysis*. Evaluation of the impact of some variation of sensors, environment and network properties, onto the performance of the WSN. Examples are fault tolerance w.r.t. sensors and network errors, effect of sensor aging or moving of their location, etc.

In this paper, due to the lack of space, we focus mainly on the problem of **planning** a WSN. To this purpose, the interactions among the modules shown in Figure 1, can be organized to define a proper design flow, as described in the following.

The user begins by formally defining the events to be captured and possibly some optimization goals/constraints and network properties, in the case of default settings are not considered suitable. Based on the existing model of the environment, the events are "fired" to get a profiling of the evolution of the physical parameters corresponding to the events, which will be used as a testbench to compare the performance of alternative WSNs. The predicate analyzer and the selected optimization goals (see Figure 1) are extensively used by the network planner to explore the design space.

3. WSN DESIGN

As said before, the model of the environment is 3-D, so that each point is represented using (x,y,z) coordinates belonging to a user-defined grid. Before starting the exploration of the WSN design space, there are three preliminary steps to define the purpose of the network, the benchmark and the hardness to recognize physical parameters corresponding to an event.

The first activity is the definition of an overall **Sensing Goal** (SG) for the WSN, that is a multi-value logic formula composed of some predicates Pr (implemented via plugins), each corresponding to an event.

For example Water(x,y,z,magn,trend) is a plugin modeling the presence of *water* in the point (*x,y,z*), starting from a given *magnitude* and with a specified *trend* over the time. A predicate Pr is an instance of Water applied to a specific point. A catalog of plugins (e.g., Fire, Water, Humidity…) is available, but its extension is straightforward.

An example of sensing goal is the following.

SG=Water (0,1,2, 20, const) AND Water (3,3,5, 10, const)

Such SG means that the WSN has the goal to discover the concurrent presence of the events of having a certain amount (20 and 10) of water in two points (0,1,2), (3,3,5) of the environment.

The second step is the characterization of the changing in the environment whenever the events occur, i.e. the identification of a **testbench** to evaluate the WSN performance. To this purpose, based on the (user defined) *fp* sampling rate of the environment simulator, a profiling stage is triggered by firing each of the defined events, namely running the Pr-related plugins. At the end, ∀(x,y,z), and ∀ Pr of SG, all the data patterns are obtained.

The other two problems the designer have to face with during WSN design, concern:

- Which types of sensor have to be chosen?
- What is the best positioning of the sensors, in order to maximize their attitude to recognize the events, i.e. maximizing the SG?

The former question impacts mainly on the feasibility of designing a WSN capable to recognize the events encompassed by the SG. The latter is related to the dissemination of sensors in order to enhance their possibility to satisfy the Pr composing the SG, i.e. improving the performance of the system.

In the current implementation of SWORDFISH, we followed an approach allowing obtaining results in the order of seconds, so to actually enable sensitivity analysis, whose discussion is beyond the scope of this paper, even if it is one of our active research topic. Thus, first of all we ensure that a solution to the SG can exists, using a proper set of sensors that is incrementally built up and significantly optimized by sharing sensors among the set of Pr (specified in the SG) to be verified. Then, this set of candidate sensors are placed in the environment taking into account the information coming from a configurable *hardness* function. In such a way it is guaranteed to obtain a WSN formally satisfying the SG with a quasi-optimal cost, with runtimes in the order of a few seconds.

As far the positioning of the sensors is concerned, we defined a *hardness* function Hard(x,y,z,Pr) modeling the difficulty in evaluating Pr in a given point (x,y,z). Calling PPr the "profiling output" of Pr, i.e. the data pattern associated to Pr obtained during the initial profiling, we define the Hard function as follows.

$$Hard(x,y,z,Pr) = Hs(PPr,t)/\ C\{(PPr,t), Pr\}$$

Where:

- Hs(PPr,t): depends on the type of sensor (corresponding model) and relates to the difficulty to recognize the event Pr within the time frame of a profiler sampling rate (1/fp). For example for a slow temperature sensor can be hard (or even impossible) recognizing T-ramps moving faster than its cutting frequency.
- C{(PPr(x,y,z), Pr} is the confidence to infer the truth of Pr based on the sequence of the physical variations defined via PPr.

Of course, any positioning strategy for the sensors attempts to place the sensor where *Hard* is low, i.e. where it is easier and reliable recognizing the Pr composing SG.

To better explain how the selection of the proper sensors take place, let us consider a simple example: three sensors (S1, S2 and S3) have been identified valuable for the four predicates P1-P4, and P4 is not yet covered by any sensor (see Table 1). Our goal is to ensure the selection of a proper set of sensors capable to cover all the predicates composing the SG.

The implemented strategy is quasi-optimal and in this case it search for a sensor among S1-S3 to sense (cover) also P4, such that its sharing produces the minimum impact onto the overall satisfying of SG, as already obtained through P1-P3.

Table 1. Sample WSN with three types of sensor.

	P1	**P2**	**P3**	**P4**
S1	X			
S2			X	
S3		X		

To support such optimization process, the operators of the SG logic formula are mapped onto a derivable expression. In particular AND and OR logic operators have been mapped onto "+" and "*" algebraic operators. In such a way it is simplified the analysis of the influence of SG w.r.t. each of the Pr composing it, by simply considering its partial derivative.

More formally, it is selected the Si to be assigned to the predicated Pj, such that $|dSG/dPj|\ \forall$ Si available, is minimum.

In the above example, we assume that S3 is the minimum (and of course it is valid to recognize the physical parameters required by P4), so that the new allocation of the sensors to the predicates becomes that of Table 2.

Table 2. Sharing of S3 among P1 and P4.

	P1	**P2**	**P3**	**P4**
S1	X			
S2			X	
S3		X		**X**

The implemented algorithm actually starts considering only the models of the available types of sensors and the predicate Pr to be satisfied, with possibly additional constraints (e.g., cost figures) that can be provided by the users within the sensor plugins. Then, the minimum set of sensors capable to recognize physical parameters to satisfy all the Pr is discovered and initially allocated to the most relevant predicates (in the SG sense). Based on this initial allocation, that is a pre-condition to satisfy SG, the sharing of the sensor proceeds as described in the above example. The end of the process produces a solution employing the minimum set of sensors covering all the predicates, using a quick heuristic producing a configuration that in most of the cases it is also the absolute optimum.

To represent how a given sensor is actually capable to capture its target events from a position (xp,yp,zp), a proper metric has been defined, called *confidence*.

$$Confidence\ (event) = 1 - (Hard\ (xp,yp,zp,Pr))/\ max\ Hard(x,y,z,Pr))$$

Where Pr is the predicate corresponding to the event, Hard (xp,yp,zp,Pr) is the hardness calculated in the candidate point for the sensor positioning and max Hard(x,y,z,Pr) is the maximum hardness within the considered environment. Note that values of confidence closer to one means that the position of the sensor is approaching the best existing in the environment to satisfy Pr, while lower values corresponds to critical points; this latter case can trigger the search for a better positioning or the increasing of the sensor set cardinality.

In the case a sensor is shared between a set of events corresponding to a group of predicate *P_set*, the confidence is calculated in the same

manner, but summing the hardness of all the predicated Pr the sensor have to cover, that is:

$$\text{Confidence (event)} = 1 - \frac{\sum_{r \in P\_set} Hard(xp, yp, zp, P_r)}{\max Hard(P\_set)}$$

The optimization strategy can be tuned to modify this default heuristic by introducing some *taboo* conditions such as a maximum number of sharing as well as some additional figures like the cost of sensors or the requirements to achieve multiple coverage of predicates to enhance fault tolerance/reliability of the WSN response. In summary, Figure 2 depicts the pseudo-code steps of the WSN planning implemented in SWORDFISH.

1. Analysis of the inputs (sensing goal parsing and constraints processing)
2. Storing of the initial condition for the environment simulation
3. Profiling of the events composing the sensing goal (storing of the data for each physical parameters and point, given an observation window and a user defined sampling rate of the simulation)
4. Computation of the hardness grid for each predicate composing the sensing goal
5. for (numSensors=1; numSensors < maxSensors; numSensors++) {
    a) choice of the target predicate for the sensors (depending on numSensors and sensing goal)
    b) computation of the sensors positions (based on Hardness and numSensors)
    c) if (check_WSN() = = OK) break }

Figure 2. Steps of the planning strategy.

As said before, the placement of sensors is based on the use of the Hardness grid obtained by adding the contribute of each of the predicates that the sensor have to recognize. In such a way, the identified position will be optimal in the sense of the minimum Hardness total value.

4. EXPERIMENTAL RESULTS

To figure out the practical use of SWORDFISH, some simple while representative examples have been selected for this paper, to emphasize the flexibility of the approach and that, even when the complexity of the application seems to be manageable, finding the optimal solution is sometimes not so "obvious".

Particular attention is paid to show the benefits of a design framework based on a formal methodology in the case of sensor sharing (to increasing the effectiveness of the WSN) and when the sensing goal is composed of a mix of physically different events.

4.1 Case 1. Sensor sharing

This case concerns the search for a WSN capable to recognize an event with a scarce amount of sensors. The user have to specify the max number of sensors, the min value of the SG considered acceptable and other data regarding the observation window for the sensors. We have chosen the following SG, corresponding to the presence of water on the ground in two positions:

SG=water(0,0,0) AND water(2,2,0), with min SG=0.2

Table 3 reports the output of SWORDFISH (sensor position) along with the confidence for each predicated. In this simple case the SG is close to one, pretty over the 0.2 threshold.

Table 3. Position reports the placement for the set of sensors.

Sens	Pos	Confidence		
		Water(0,0,0)	Water(2,2,0)	Total
S0	(1,2,0)	0,999	0,999	0,998

The truth value of the single predicates are similar and close to one (water(0,0,0)=water(2,2,0)=0,99999) so that the SG=0,999 is fairly acceptable. In summary, the system discovers an intermediate position for a single sensor (1,2,0) ensuring the meeting of the SG with sensor sharing. In the case our goal is modified to recognize the presence of water in two points more distant as above and with an observation window of 5 seconds, i.e.:

SG=water(0,0,0) AND water(9,9,9), with min SG=0.2

The system fails in using only one sensor and find out automatically a new WSN using two sensors, now satisfying the SG. Because of we have two sensors for two events, the suggested positioning of the sensors are obviously overlapped to the event locations (Table 4).

Table 4. New solution with two sensors.

Sens	Pos	Confidence		
		Water(0,0,0)	Water(9,9,9)	Total
S0	(0,0,0)	0,9999	0,	0,999
S1	(9,9,0)	0,1	0,9999	0,999

Note that Table 4 highlights a (negligible in this case) contribute of S1 also to water(0,0,0) recognition. Such type of information can be useful to identify *Achilles' heel* of more complicated WSNs, where the amount of sensors makes hard identifying their ordering of relevance in contributing to the overall SG.

4.2 Case 2: Influence of the type of event

The WSN we are designing has the responsibility to report the presence of two different events (water and fire) having different sensing requirements. In particular, in our modeling environment sensing the water it is harder than recognizing the fire. The SG is the following:

SG=water(0,0,0) AND fire(5,5,0), with min SG=0.4

In this case, as shown in Table 6, the positioning of the sensor is closer (0,3,0) to the water, because of the sensing of fire is considered easier than recognizing the water itself. In more complicated scenarios, but

even in this simple case, the typical design approach to place the sensors in intermediate positions disregarding the type of events to be considered, does not allows to take full advantages from the WSN intrinsic capability.

Table 5. The type of event influences the sensor placement.

Sens	Pos	Confidence		
		Water(0,0,0)	Fire(5,5,0)	Total
S0	(0,3,0)	0,99	0,99	0,99

5. SWORDFISH SOFTWARE SYSTEM

The entire SWORDFISH software system has been developed in C under GNU/Linux (Debian distribution), using the following libraries:

- XanLib library ver. 1.1: to manipulate data structures like hash tables, trees, pipes, etc.
- Gtk library ver. 2.0: for the GUI development.
- GNU C library ver. 2.3: to interface with the GNU system.
- Flex: to make a lexical analysis of the user's input describing the sensing goal.
- Bison: to generate a parser for the grammar which describes the multi value logic used to express the sensing goal.
- Libglut: for writing the world in a 3D vision.
- Libgtkglext: to embed the openGL objects inside the GTK GUI of SWORDFISH.
- Graphviz: to draw the direct graph representing the sensing goal written by the user and each of its derivatives.

The software architecture is composed of the following main modules: Simulator; Planner; Logic_manager; Sensors; Events.

Simulator has the goal to emulate the behavior of the supported physical events (e.g., a fire or an atmospheric phenomenon), while the *Planner* has the role to design the WSN by satisfying the input constraints.

A crucial module is the *Logic_manager*, capable to manage the multi-value logic on which is sitting the writing of the sensing goal and of the constraints. Such a module is invoked by the Planner to calculate the truth of a predicate, as a result of certain spatial distribution of the sensors, as well as to calculate the derivative of the logic functions.

Sensor is the manager of the sensor models implemented in SWORDFISH. It is realized via a standard interface based on dynamically loadable shared libraries (plugins). Thanks to this choice, it is allowed to manage efficiently a wide range of sensors with no impact on SWORDFISH code, since the only contact is through the functionality exposed by the interface. Sensor models can be added incrementally as well as obtained interfacing other libraries, without changing SWORDFISH software.

Events is in charge of managing the implemented available types of events. Its implementation is similar to Sensor, since it uses plugins to decouple the implementation of the events from the rest of the SWORDFISH software.

The current version performs the design of WSNs with a dozen of predicates composing the SG and a similar amount of sensors with runtimes of less than a minute, running on a 1.8 MHz Centrino Laptop. Bigger WSNs (tens of predicates and sensors) requires 1-2 minutes to produce the result. The execution time of SWORDFISH is considerably influenced by the time window chosen by the user.

6. CONCLUSIONS

The paper presented SWORDFISH, a framework to support the design of sensor networks. In particular, the focus has been on the methodology and design flow to plan WSNs, although SWORDFISH toolset encompasses also the possibility to perform verification and sensitivity analysis of existing WSNs.

The obtained results are promising and with simple representative examples it has been shown how it is possible to optimize the WSN while formally ensuring that the original user' goal has been fulfilled. The examples reveal that many side-effects of changing the behavior of the WSNs and sensor positioning produces strong modifications on the sensing goal that are hard to be managed by a human designer, without the support of a proper tool, since frequently they are not "intuitive".

In summary, the presented approach is complementary to the typical simulation-based analysis frameworks, since its emphasis is more on the system-level steps of the design, where a broad design space has to be extensively and efficiently explored, and on the formal modeling and verification of the WSN objectives.

Work is in progress as part of a large multi-partner project**,** where the biggest test case is a high quality wine producer, where a set of wide vineyards as well as the grape processing, have to be monitored according to a number of parameters and critical conditions to be avoided. In such a case, reliability, aging of the components and sensitivity to the sensor positions have also to be considered in detail.

REFERENCES

[1] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cyirci, E., Wireless sensor networks: A survey. Computer Networks, 38(4):393–422, 2002.

[2] TOSSIM: download from www.tinyos.net

[3] NS-2: www.isi.edu/nsnam/ns/

[4] Avrora:research.cens.ucla.edu/projects/2006/Systsms/Avrora

[5] J-Sim: www.j-sim.org

[6] SENSE: www.ita.cs.rpi.edu/sense/

[7] OMNeT++: www.omnetpp.org

[8] VisualSense: ptolemy.eecs.berkeley.edu/visualsense

[9] SensorSim: nesl.ee.ucla.edu/projects/sensorsim/

[10] EmStar: cvs.cens.ucla.edu/emstar/

[11] OPNET: opnet.com/products/modeler/home.html

[12] J.Polley, D.Blazakis, J.McGee, D.Rusk, J.S.Baras, and M. Karir, *ATEMU: A fine-grained sensor network simulator*, in Proceedings of SECON'04, First IEEE Communications Society Conference on Sensor.