Chapter 19
Optimal Design of Wireless Sensor Networks

Marcello Mura, Simone Campanoni, William Fornaciari, and Mariagiovanna Sami

Abstract. Since their introduction, Wireless Sensor Networks (WSN) have been proposed as a powerful support for environment monitoring, ranging from monitoring of remote or hard-to-reach locations to fine-grained control of cultivations. Development of a WSN-based application is a complex task and challenging issues must be tackled starting from the first phases of the design cycle. We present here a tool supporting the DSE phase to perform architectural choices for the nodes and network topology, taking into account target performance goals and estimated costs. When designing applications based on WSN, the most challenging problem is energy shortage. Nodes are normally supplied through batteries, hence a limited amount of energy is available and no breakthroughs are foreseen in a near future. In our design cycle we approach this issue through a methodology that allows analysing and optimising the power performances in a hierarchical fashion, encompassing various abstraction levels.

19.1 Introduction

When envisioning applications that directly interface with the physical world through use of wireless sensor networks (WSNs), some low-level design aspects have to be taken into account already while drafting the application at a high abstraction level. In fact it becomes necessary to consider some requirements and constraints that go

Marcello Mura
ALaRI - Faculty of Informatics - University of Lugano (CH)
DEI - Politecnico di Milano (IT)
e-mail: muram@usi.ch

Simone Campanoni · William Fornaciari · Mariagiovanna Sami
DEI - Politecnico di Milano (IT)
e-mail: {campanoni,fornacia,sami}@elet.polimi.it

G. Anastasi et al. (Eds.): Networked Enterprises, LNCS 7200, pp. 375–395, 2012.

well beyond the ”typical” information technology ones: just as an example, a few points may involve:

- The physical environment where the sensor network will be deployed. Dimensions are not the only relevant point; topography, presence of physical obstacles, even such aspects as availability of sunlight may have an impact on design or even on feasibility itself;
- Requirements of sensor deployment (e.g., critical positioning of some given types of sensors);
- Technological choices (e.g., are there commercially available devices capable of sensing specific phenomena? Are such sensors capable of being ported into a sensor network in terms of costs, dimensions, energy consumption?);
- The ”survival” requirements for the sensor network. If (as it is usual) battery-operated nodes are envisioned, how long do we expect the network to operate? Is this goal compatible with technological choices as well as with processing and transmission loads?
- How harsh is the physical environment and how critical is survival of an individual node and of the network as a whole?

Only after such basic questions have been answered it is possible to state whether (given the available technology) an efficient and effective deployment will be possible and what is necessary in order to achieve it. To this end, the application designer must be able to rely on a full design flow, based on suitably developed methodologies and tools, that will ultimately support the low-level design but that will also provide a first set of indicators guiding high-level decisions. More specifically, the designer should be able to:

- Verify the feasibility in terms of basic technologies;
- Take preliminary decisions (e.g. , communication protocol, programming model, etc.);
- Evaluate costs, power consumption, capacity of survival of a first draft design and check them against the specifications;
- Identify the aspects for which optimisations might be particularly relevant;
- Provide low-level specifications for the final network design.

Coming to the specific application envisioned within ArtDeco, recurring to sensor networks in fine-grained monitoring of high-quality cultivations has great potential relevance, as it allows bringing integrated adoption of ICT techniques from the initial production to the final commercial support in a sector (the so-called primary sector) where such overall impact has not yet been fully exploited. Remote sensing techniques, adopted by a few very large actors, are rather costly, do not provide 24-hours coverage and afford limited precision with respect to micro-climate aspects and to some ground-level sensed data. Use of sensor networks would overcome such limitations and make advanced monitoring available to a much larger community of users.

The experiment envisioned for ArtDeco represents a spectrum of applications in which sensor networks must be deployed in a well-studied and carefully tuned way, rather than by random distribution of highly redundant numbers of nodes. Position

of the individual nodes is related to physical characteristics of the environment, and this will in turn impact on network operation. Cost needs to be minimized. Analysis and optimisation must thus be performed for the specific network well before actual deployment to support deployment decisions.

The initial design phases of a WSN require modelling and simulation tools meeting a number of challenges: as different aspects of the system need to be analysed, different modelling approaches may be more suitable and different components of the system may be targeted (often attention is on the radio section, but processors and sensors as well may need specific attention). Hence, identifying a single reference tool satisfying the requirements of a vast majority of designers is difficult.

Various approaches to deal with this task have been proposed, and related simulation tools have been made available (see, e.g., [1], [4], [14], [15]); nevertheless, custom-built simulators may need to be realized in specific instances (see, e.g., [7] where the necessity of modelling the node together with its harvesting section did not allow the use of a standard network simulator).

A major research focus is represented by optimising the location of the sensors to maximize their collective coverage of a given region. This challenge has been tackled using several approaches, such as integer programming or greedy heuristics to incrementally deploy the sensors. Adaptive techniques considering scenarios where multiple sensors are needed, accounting for a possible real-time deployment have been proposed.

Functional and non-functional requirements of a WSN may be very strict and must be tackled at design time. Power-aware design and operation of nodes as well as of network has received in particular much attention. Several power-aware protocols and routing algorithms have been presented (see, e.g., [3], [6]), the ultimate goal being that of increasing battery and network life.

A further factor that may have great relevance in some application classes is the capacity of prompt sensing of, and reactions to, particular events. The planning of WSNs for such applications is a off-line activity and requires to:

1. specify the characteristics of the events to be discovered;
2. select a proper set and type of sensors to enable the capturing of such events;
3. embed the sensors in the environment in a way to ensure the capturing of the desired events while optimising some design goals.
4. estimate energy consumption of proposed solutions so as to evaluate their viability in terms of network lifetime and eventually define optimisation policies

The approach described in this chapter is first of all to make sure a priori, with a good confidence, that there exists a feasible solution to the sensing problem with the accuracy required by the application. Then, by exploiting the capabilities of the SWORDFISH optimisation engine, the WSN is refined according to design constraints and users goals. After a preliminary network has been drafted, power modelling is tackled following a hierarchical approach (see [12]) that allows creating technology independent models at a high abstraction level and subsequent estimation and comparisons when low level technology-dependent models are inserted. Node and network level simulations based on such approach allow to refine the various choices

(from node selection to network organization) and to perform possible power-related optimisations.

This chapter presents the contributions given by the authors to the design flow of a WSN, concerning both the initial phase of node-level and network-level design in view of a specific application and the subsequent steps focusing on power-oriented simulation and optimisation. The chapter is organised as follows. The concept of multi-level WSN design methodology is introduced first (section 19.2); design tools developed for sensor-level, node-level and network-level design are presented in the subsequent sections 19.3 to 19.6. The methodology devised for modelling power consumption at node and network level is then discussed in sections 19.7 to 19.9. A case study is discussed in section 19.10.

19.2 Multi-level Design

One of the objectives of the multi-level design methodology is to create a design flow for WSN offline planning, which is scalable with the application complexity (see Table 19.1). To this purpose, the first step is to identify a proper set of sensor-position pairs, considered optimal to capture the desired behaviour of the WSN.

Table 19.1 Abstraction layers of the design space exploration

Level	Activities
Sensor	Selection and positioning of sensors set; Sensitivity analysis
Node	Aggregation of sensors onto some nodes; Sensitivity analysis
Network	Identification and positioning of gateways; Protocol selection

This initial solution is the baseline for any architectural design space exploration. The next optimisation step is related to possible aggregation of the previously identified sensors set onto nodes, so as to balance network cost with effectiveness and performance of the WSN. The outermost layer accounts for complex sensor networks or for networks deployed in rugged terrains, where some of the nodes have to manage hierarchies of sensors/subnets.

We consider here all of the three levels mentioned in Table 19.1. The support to such system-level design is provided by a modular framework, called SWORDFISH. 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 optimisation parameters), whose roles are explained below and depicted in Figure 19.1.

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

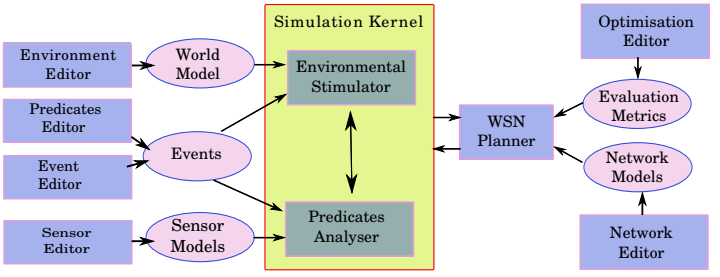

Fig. 19.1 Coarse-grain architecture of SWORDFISH

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 allows obtaining the analytic representation of the sensing nodes, which is a modelling 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 when a given event occurs. This is of paramount importance to verify that a WSN is actually capable to properly react when 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 modelled and verified by the Predicate Analyser (see Figure 1).

Optimisation 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 formally verifying that a given WSN is able to capture a set of events as well as to support the building and optimisation of the overall network according to the selected policies and goals.

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: the goal is to 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.) by exploiting the potential of a given WSN

Sensitivity Analysis: evaluation of the impact of some variations of sensors, environment and network properties, onto the performance of a WSN. Examples are fault tolerance w.r.t. sensors and network errors, effect of sensor ageing or displacement of their location, influence of the observation time, etc.

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.

Based on the application requirements, the first step for the user is formally defining the events to be captured and possibly some optimisation goals/constraints (Predicate, Event and Optimisation editors). Network properties and sensor behaviour can be also specified (Network and Sensor editors), if the default settings are not considered suitable. According to the model of the environment (specified using the Environment editor), the events are then *fired* to get a profiling of the evolution of the physical parameters corresponding to the events. Such results are then used as a testbench to compare the sensing capabilities of alternative WSNs. The Predicate analyser and the selected optimisation goals are extensively used by the Network Planner to explore the design space. Useful information for optimisation can be gathered by analysing the sensitivity of the network over the variation of parameters like observation time or sensor accuracy.

19.3 Sensor-Level 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 of recognizing 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 (implemented via plug-ins), each corresponding to an event. For example Water(x, y, z, magn, trend) is a plug-in modelling 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 is an instance of Water applied to a specific point. A catalogue of plug-ins (e.g., Fire, Water, Humidity) is available, and its extension is straightforward. An example of sensing goal is the following.

$$SG = Water(0,1,2,20,const) \wedge Water(3,3,5,10,const)$$

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

The second step consists in characterizing changes 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) sampling rate of the environment simulator, a profiling stage is triggered by firing each of the defined events, namely running the related plugins. At the end, for every (x,y,z), and for every predicate of SG, all the data patterns are obtained.

The other two problems the designer has to face concern the types of sensor to be chosen and their best positioning, in order to maximize their capacity to recognize the events, i.e. maximizing the SG. The former point impacts mainly on the feasibility of designing a WSN capable of recognizing the events encompassed by the SG. The latter is related to the dissemination of sensors in order to enhance the possibility of satisfying the composing the SG, i.e. improving the systems performance.

SWORDFISH is a very fast tool, able to provide results within seconds of computation, so as to actually enable sensitivity analysis. 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 optimised by sharing sensors among the set of (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 that a WSN formally satisfying the SG with a quasi-optimal cost will be obtained, with runtimes in the order of a few seconds.

As far the positioning of the sensors is concerned, we defined a *hardness* function modelling the difficulty in evaluating in a given point (x,y,z). A formal definition of hardness and confidence can be found in [2]. The hardness is a function linking: (i) the data patterns obtained during the initial profiling (depending on the type of sensors); (ii) the difficulty to recognize the event predicates Pr (those composing the SG) within the time frame of a profiler sampling rate; (iii) the confidence to infer the truth of Pr based on the speed of variation over the time of the above data patterns obtained during the initial profiling. To represent how a given sensor is actually capable to capture its target events from a position, a proper metric has been defined, called confidence [2], that is comparing the hardness in one point, with its maximum value.

The optimisation strategy uses some default heuristic (alternatively, some taboo conditions,such as e.g. a maximum number of sharing may be imposed). Additional features like the cost of sensors or the requirements to achieve multiple coverage of predicates to enhance fault tolerance/reliability of the WSN response may be considered.

As mentioned above, the placement of sensors is based on the use of the Hardness grid obtained by adding the contribution of each of the predicates that the sensor has to evaluate. In such a way, the identified position will be optimal in the sense of reaching the minimum Hardness total value.

19.4 Cost Model

The sensor-level design, carried out within SWORDFISH, produces a set of (sensor, position) pairs tailored to optimise cost-effectiveness and capability to fulfil the sensing goal of the WSN. On the other hand, realistic design and deployment typically require simplifying both node hardware and network architecture , by exploiting boards hosting multiple sensors. This constraint necessarily modifies the optimal positioning of sensors, with the risk of side effects on the desired WSN behaviour.

Depending on the application, three different cases can be envisioned: new ad-hoc boards are realized for the application, use of off-the-shelf boards already existing on the market, and customization of boards, e.g. by adding daughter boards to create gateways or to add specific sensors.

Based on market availability of sensing modules and the results emerging from the application scenarios defined in ARTDECO research project, we found reasonable to adopt a general model of monetary cost for each board (node). We observed that there exist a variable cost which is related to the type and number of sensors in a linear manner and a processing cost that is logarithmic, due to the typical price trends of CPU and micro-controllers.

To consider different suppliers, we partitioned the available sensors into classes, to capture their relative cost, instead of considering the absolute values. Concerning the cost of the network, we assume a constant value depending on the protocol for wireless connections (typically built-in in commercial nodes).

Furthermore, some influence of the network topology should be considered in the case of some gateway nodes, managing hierarchies of sensors patches, were identified. In such case, there is an additional cost related to the wired connection or the use of other long-range radio communication standards and modules.

19.5 Node-Level Design

The clustering of the set of sensors identified by SWORDFISH is a multi-stage process, including the following main activities : compatibility analysis between all the possible pairs of sensors, identification of the boundaries of the clustering problem (worst and best case), and generation and evaluation of the candidate solutions.

Compatibility. Initially, the user (e.g., by accepting default settings) has to provide taboo conditions, by specifying constraints on the possible clustering of different sensors onto the same board. Based on these information, an Interference Graph $G=<N,E>$ is built, where nodes n are sensors and an edge e between two nodes represents a possible sensor interference to be avoided.

From the interference graph, the complementary compatibility graph $G'=<N,E'>$, gathering all feasible solutions, is built. (Note: any possible clustering of sensors is a clique of the compatibility graph, since all sensors hosted by the same board must be compatible with each other). All the maximal cliques of the compatibility graph G'

is computed next; since this is recognized to be a NP-hard problem, some heuristics are adopted.

Coverage. At the end of the previous step we obtain a partitioning of the compatibility graph in cliques clustering the maximum number of compatible nodes. The design space spanning between the two boundaries cases so identified contains a number of possible solutions that is exponential with the sensor cardinality. Suitable heuristics are introduced to extract a set of (not necessarily maximal) cliques, allowing the optimiser to consider solutions possibly less homogeneous but characterized by a lower board-level cost.

Comparison of Solutions. This step takes into account the candidate WSNs from the Pareto standpoint. The task of the *Pareto Efficient Solution Clustering Algorithm* (PESCA) is to find out a solution to the multi-objective clustering problem, considering two metrics: cost and functional quality, i.e. its performance.

The cost of a solution (set of boards) is evaluated through the cost model described in Section 19.4, that is depending on the number and type of sensors associated with each partition. Concerning the performance, the quality of a solution is computed by exploiting the Hardness functions of the event i covered by the sensor j belonging to the same board. The hardness of the WSN is evaluated, and its minimum corresponds to a point where the positioning of the board is optimal. This new location, which is shared by all the sensors on the same node, is the best to ensure that all the events associated with the sensors can still be captured after clustering. The solution so discovered is a Pareto efficient solution.

19.6 Network Design

In the previous sections, node level analysis was considered. It is anyway useful to move further our perspective, and to include network dimension in the analysis/optimisation phase. Problems such as network topology, gateways placing, definition of a suitable communication protocol or tuning of an existing one must be addressed considering the particular application.

Fine-grained monitoring of a high-quality vineyard may require deploying a consistent number of nodes, especially if the area to be monitored is wide. If the monitoring area is small or at least more uniform with respect to the measured parameters, networks can be smaller and simpler. It is possible to group the network topologies in two categories.

Star Topology. The star topology is the simplest network topology we consider; a central coordinator is responsible of orchestrating network activities so as to collect information from sensor nodes. All the nodes in the network have a communication link only with the coordinator. In the particular case of the precision agriculture system of Donnafugata it is convenient to use such a schema when the distance between nodes is small enough that there exists a radio link between all the nodes and a

central coordinator. Given current legislation limits[1] the range should be in the order of hundred of meters to few Kilometers; considering power requirements and current technologies the range should be reduced to tens of meters to one hundred meters for most applications.

Multi-hop/Mesh or Hierarchical Topologies. There may be cases in which extension of the deployment area of the WSN (or the location topographical characteristics) does not allow the use of the star topology. It may be then necessary to send messages with data from sensor nodes stepping through a set of intermediate nodes before reaching their final destination (i.e. multi-hop). Alternatively, it is possible to have local star networks with a gateway nodes, while the information is locally exchanged using a star topology, gateways communicate between them (usually through a wider range network) forwarding or possibly aggregating information coming from the node.

The problem then requires a design space exploration phase that allows identifying the appropriate solution for the particular application class targeted. The design space can be seen as made of multiple dimensions; as an example, choosing or tuning the communication protocol involve evaluation of a multiplicity of parameters; moreover topological choices as optimal partitioning of the network or selection of coordinators, gateways or sinks should be considered.

While in the case of medium/small size WSN it is possible to identify the optimal solution, as the network grows heuristics must be used producing near optimal solutions. In this phase, the positions of the nodes, evaluated in the previous phases, are taken as an input to calculate the optimal routing through the network and the parameter configurations while keeping the nodes positions fixed.

The output of this phase is a set of Pareto-efficient solutions, specifying: the type of the nodes, the partitioning of the network (in terms of communication links), the overall throughput of the WSN, and the cost of the WSN.

19.7 Power Modelling Methodology

The previously described design phase tackles functional requirements and network cost in terms of components and placement so as to reach a first feasibility assessment. A second main problem remains to be tackled, namely, dealing with energy efficiency and power limitations. Power consumption cannot be navely inserted in the cost functions used above, as it is related to network operation; the previous choices obviously impact on it, and ultimately the power analysis phase may lead to modifying either choices for nodes or network topology or possibly even network protocol. A further optimisation phase then has to be carried out, based on suitable power-related modelling and simulations at node and network level.

[1] The maximum power considering the band and the modulation for 802.15.4 networks is 13dBm.

We approach power modelling for WSN in a classical top-down way, starting from abstract choices such as the adopted protocol, the generic node and network structure, etc.. An abstract, implementation-independent model thus derived then allows both validating the high-level concepts and proceeding through subsequent design space exploration steps, by identifying critical points, possibly suggesting optimisations that still comply with the initial solutions, evaluating and comparing from the energy perspective the alternative implementations, supporting feasibility decisions for a specific application.

The modelling style chosen is that of StateCharts [5], a well-known and widely adopted formalism that allows us to efficiently model hierarchy and concurrency and to detail operation of Logical Activities. At the highest abstraction level, the application is modelled through a (set of) FSMs representing the behaviour of the system; no implementation choice is evident here. Operation of this FSM corresponds to activation of lower-level, concurrent FSMs representing with finer detail solutions that have been adopted in design; consider a very simple example, referring to the Donnafugata case study and to the individual node model. The end-user and the designer initially reach an agreement on the node operation, from which the top-level FSMs representing the nodes operation are derived (see figure 19.2 A)). No information on actual low-level actions, technologies, protocols etc. is as yet present. Operation of this FSM can be seen as a path in the state diagram, traversing suitable nodes in suitable order; reaching a state in this FSM actually activates one or more lower-level FSM, that represent an (intermediate) implementation. When creating and activating the lower-level FSMs, design detail is added and choices are made (e.g., concerning the transmission protocol), but technological detail may still be absent (see figure 19.2 B)). For example, one of the the top FSMs includes a sense state, which at lower level activates a path on the FSMs coordinating the behaviour of the individual sensors and of the microprocessor, even though no technological detail on any of these components is as yet introduced (as an example the number of sensors is yet not relevant at this abstraction level). Iteratively, design detail is added by creation and activation of lower-level, concurrent FSMs. Stepping further down, execution of the chosen protocol is modelled as an activation of elementary machines representing the behaviour of all the different components of a node, and finally adding detail to such machines until the physical implementation (bottom) level is reached, where actual values (in particular, concerning energy and power) are introduced to annotate the bottom FSMs. A similar procedure holds at the network level, where the initial deployment reached through use of SWORDFISH may be taken as the starting point.

Defining a possible high-level use of the nodes implies subsequent FSM activations through the whole hierarchy and finally provides power consumption estimation for that use mode; modelling a different use mode will just result in different firing of lower-level FSMs, while exploring different choices will again lead to different firing sequence and/or to different bottom-level power annotations.

As already said, at the bottom level the technological details are taken into account so that quantitative information may be added in correspondence of a specific implementation. Once more, we start from an abstract (implementation-independent) model representing providing functional but not technological detail, afterwards

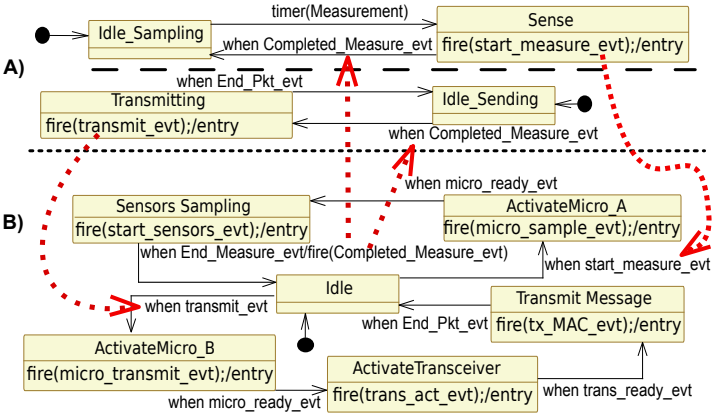

Fig. 19.2 A simplified example showing the mechanism of hierarchical modelling

annotating it with implementation-specific information. To this end, in order to analyse the nodes energy consumption we identify first the different sources of consumption, performing both an architectural and a functional breakdown. Keeping the model implementation-independent means that, rather than detailing a specific hardware architecture, execution of a protocol on an abstract node architecture is modelled. Power consumption is related to both the functional sections of the architecture (i.e., transceiver, processing unit, sensors etc.) and to the activity performed by it. This follows the concept of *Logical Activities (LAs)* introduced in [12] that leads to estimating the total energy consumption as the sum of the energies corresponding to the sequence of activities performed by the node (the implicit linearisation assumption has been experimentally proved to grant acceptably accurate results). The abstract architecture includes the communication standard and models the related timing information as well, either by adopting definite values (whenever strict timing constraints are given) or by inserting *Temporal Parameters (TPs)* whenever flexibility of timing is allowed.

Starting from the abstract architecture, an *Implementation Independent Model* is built; this model can then be characterized for a specific platform obtaining an *Implementation Specific Model*. In this final modelling, quantitative information is associated with the components of the (purely qualitative) abstract model derived before. Correspondence to operation of actual platforms validates both Implementation Specific and Implementation Independent models.

The methodology comprises the following phases:

- **Abstract Model:** Starting from the protocol and the application, LAs and TPs are identified.
- **Implementation Independent Model:** the FSMs model of the system is built based on the Abstract Model,. *LAs* are mapped onto states (and when necessary to transitions) and *TPs* are inserted as timeouts on the corresponding transitions of the FSMs.

- **Implementation Specific Model:** By performing a series of measurements, the implementation-independent model is characterized for one or more given platforms. In particular, in this phase the power values corresponding to the *LAs* and the times corresponding to the *TPs* isolated in the previous phases are evaluated.
- **Experimental Validation:** The Implementation Specific Model is evaluated by comparing its predictions for a set of activities of the nodes to the measured consumptions of the node. (Iteration of some previous phases may be requested to achieve a more accurate model).

19.8 Hierarchical Modelling

The low-level node model consists of independent FSMs associated, respectively, with radio, microprocessor and sensors. These FSMs are then suitably activated and coordinated by the higher-level operations required by protocol execution and ultimately by the application[2].

19.8.1 Bottom Layer Machines

To better clarify the LA concept, let us refer to an example: if we are considering the transceiver activities, we may isolate different sources of Power Consumption, e.g. *Reception*, *Transmission*, *Idle*, *Low-Power*. We extract such LAs in an abstract way by analysis of the communication standard.

Identification of LAs represents the basics for creation of the bottom layer machines. We only annotate power consumption in lowest-level state machines; in such FSMs there is a one to one mapping between States and LAs. Subsequently, designers using the model thus created can easily see the projection of high-level actions onto activations of suitable paths in the low-level FSMs, and thus possibly identify critical points as far as power consumption is concerned. In figure 19.3 a simplified sets of FSMs suitable for the three main sections (i.e. transceiver, microprocessor and sensor) are shown.

19.8.2 Higher-Level Models

The bottom layer FSMs are driven by models at higher levels so as to compose the set of activities performed by the system. By the "event broadcast" mechanism of the StateCharts formalisms, the various layers of the system can model the behaviour of the node. Hierarchical modelling allows representing the high-level aspects of the

[2] The underlying philosophy is similar to that adopted in the case of instruction-level power modelling for microprocessors (e.g. [13]).

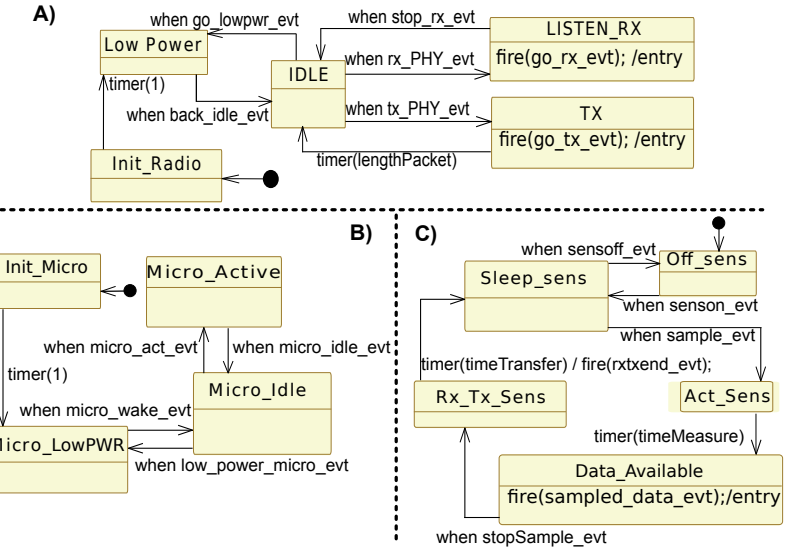

Fig. 19.3 The bottom level State Machine for the Transceiver (A), the Microprocessor Unit (B) and a Sensor (C) are shown in this figure

envisioned application with minimal or no reference to the underlying technology (i.e. technology independence).

As already hinted, at the highest abstraction level, the application (the "scenario") is modelled by an FSM that will activate the lower-level FSMs through the suitable hierarchy. At any abstraction level, FSMs operate concurrently. Considering now the hierarchy involved, on the radio side the scenario machine directly operates through a sequence of FSMs representing, as an example, scheduling of packets, networking MAC layer etc, down to the lowest-level radio model. In the same way, where the microprocessor is concerned, the sequence of FSMs corresponds in order to the scheduling and the processing of the MAC layer, to processing and storing of the sensed data, etc. The case is much simpler for the sensors section, where basically only the lowest-level FSM exists and is directly activated by the scenario FSM.

In any case, the events notified by higher level machines are not strictly bound to a particular implementation, or even to a particular choice on lower levels (e.g. choice of the MAC layer of the communication protocol etc.) but can be reused in the design space exploration phase, activating alternative lower-level FSM sets or the same FSMs in an alternative way.

In Fig. 19.3 A) a reference application running on a wireless node is modelled. The node samples a physical quantity with periodicity defined by ; the measure is added to a data packet and when a number of measures defined by are collected, the packet is sent. As the scenario starts the sensor is turned on (through senson_evt); when lower level machines notify an event communicating that the node has been correctly associated with a network (association_completed_evt) , the sampling procedure begins.

In Fig. 19.3 B) a simplified MAC layer is presented. In the top part a schematic representation of the scanning procedure is modelled; after completing scanning and association (association_completed_evt) the node synchronizes with the chosen coordinator and enters the macro-state *beaconed_network*. Operation in this macro-state is periodically reactivated through a timer so as to mimic the beacons periodic structure;

The microprocessor is restored and it wakes up the radio after a time interval - depending on the duty cycle of the network - before the beacon is received (beacon_evt notified) so as to account for listening tolerances. After the end of the beacon packet, the node enters in the macro-state corresponding to the contention access period. When the node is in the Active state it notifies it to the scheduling layer. We emphasised the Txbeaconed state so as to show how it drives the lower layer models by causing a change of state in the radio FSM model.

Further features, such as, e.g., channel arbitration, can be easily introduced even at a later time as intermediate layers in the hierarchy.

Messages are produced by the application, but nothing guarantees that the MAC layer is in a consistent state to send such packets when created. Therefore a scheduling FSM, as sketched in figure 19.3 C) is necessary. Such FSM on one side collects the requests for sending packets from the application an on the other monitors the capability of the node to transmit packet so as to notify the event triggering transmission in the appropriate moment.

Experiments carried out through a measurement campaign [9] show that results of the model approximate real consumption within an interval of less than five percent. Moreover all the main components of the nodes are included and this allows giving a real estimation of the node consumption and may help isolating some inefficiency of the protocol.

19.9 From Node Models to Network Models

In general, to allow dealing with the complexity of systems with increasing dimensions, the node models used for network simulation are more simplistic than those for node level simulators. Simplification of the model may anyway lead to missing some important aspects for evaluation and optimisation.

On the other hand node level models allow devising only some possible optimisation steps for design of the network as a whole, so that network level models are certainly needed. It is possible to emulate network level simulations by feeding various node models with synthetic data and extrapolating the performance of various solutions, but data collected in this way may not be realistic. The use of network level models gives higher credibility to the evaluation as the node model is fed with more realistic input streams.

We propose a framework for automatic generation of power simulators starting from the protocol-level model designed through our methodology. Our approach consists in generating executable C++ code, using an appropriate simulation library,

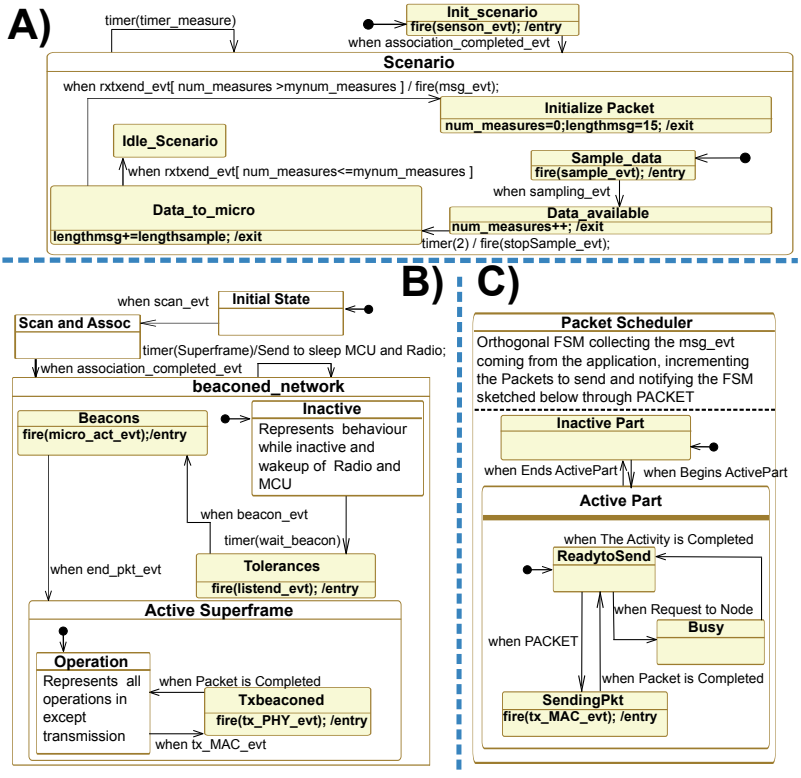

Fig. 19.4 The simplified FSMs representing the real-world application scenario A),A simplified representation of an 802.15.4 like MAC layer B) and a scheduler interconnecting Application and MAC C)

starting from the representation of the models through StateCharts formalisms [8], [10], [11]. This simulation framework has been extended to support simulation of network level models.

Multi-Instantiation of the node models is the basic step that allows building of network models. Just as in the Object-Oriented paradigm classes can be instantiated into objects, we use instantiation of models into simulation objects. Coexistence in a network simulation scenario of instances of different models may be necessary; as an example in the case of a sensor network based on IEEE 802.15.4, different models may correspond to coordinator and simple devices. The network models are built using the nodes models so that the same level of detail is kept.

As the simulation objects are executed concurrently in a common scenario, a communication pattern between various simulation objects is necessary; to this end, appropriate interfaces suitable for allowing communication between objects are defined in the various models. As the nodes are composed in a network scenario, it becomes necessary to separate a local and a global context; for example, an event that brings the transceiver of a node in receiving state is local to that node, while an event that

Table 19.2 Optimal position and type of sensors

Sensor	Position (x,y,z)	Type	Sensor	Position (x,y,z)	Type
s_0	$(0,0,0)$	Pressure	s_1	$(0,1,0)$	Temperature
s_2	$(1,1,0)$	Water	s_3	$(2,1,0)$	Water
s_4	$(2,2,0)$	Temperature	s_5	$(1,0,0)$	Pressure
s_6	$(3,2,0)$	Water			

synchronizes the network has to be processed by all the participating nodes and has global scope. Solutions dealing with this issue depend also on the network topology adopted. Our tool supports creation of network models for the two basic networking topologies presented in section 19.6 (essentially, the ones taken into account when devising the ArtDeco experiment).

19.10 Case Study

Synthetic applications and some real use cases extracted from the ARTDECO project are here used in order to provide an evaluation of the algorithms and approaches presented within this chapter.

19.10.1 Node-Level Optimisation

To highlight the importance of a quantitative tradeoff when moving toward realistic deployments, let us consider an application setup with the following sensing goal SG:
$SG = Pressure(0,0,0) \wedge Temperature(0,0,0) < 30 \wedge Water(1,1,0) \wedge$
$Temperature(1,1,0) > 20 \wedge Water(2,2,0) \wedge Temperature(2,2,0) > 20$

For the sake of clarity, the environment is open space, the sensor model is ideal, the time window is set to 1 second and there are no taboos specified.

In general, each board hosting sensors includes the following sections: PCB/package; power supply and energy management; radio (RX/TX); control/processing Unit (CU); connectors/Interfaces; one or more sensors. Based on our experience in realizing PCB-level embedded systems and on market availability of sensing modules, we found reasonable adopting the model 19.1 for the cost of each board (node).

$$NodeCost = Const + K * log(N) + \sum_{j=1}^{SensorTypes} SC_j * NumS_j \qquad (19.1)$$

Where, for each board, N is the overall number of sensors, $NumS_j$ is the number of sensors of a given type j, SensorTypes is the number of possible types of sensor, and SC_j is a cost of a sensor of type j. More details can be found in [2].

The other parameters of the cost are Const=12.5, K=0.5 and all the sensors have the same $SC_j = 1$, no matter their type. The output of SWORDFISH is a set of 7 sensors (see Table 19.2). Starting from this configuration, PESCA (see Section 19.5) computes the following two cliques with the max cardinality: $\{s_0, s_1, s_2, s_3, s_4, s_5\}, \{s_4, s_6\}$. Then the covering of G' is performed using the subgraphs: $\{s_0, s_1, s_2, s_3, s_4, s_5\}$ and $\{s_6\}$. Due to space limits, the entire set of solutions generated and evaluated is not reported. In this example there is only a single solution in the Pareto frontier, which is the following.

Sol_1 Board0=$\{s_0, s_1, s_2, s_3, s_5\}$, position=(1,1,0),
Board1=$\{s_4, s_6\}$, position=(2,2,0),
Total cost=34.8, hardness=41

Should we consider a different technology with Const=2.0 instead of 12.5, the solutions populating the Pareto frontier become those depicted in Table 19.3. It worth nothing that these solutions require more boards w.r.t. the previous one, as a consequence of the reduction of the board model fixed cost. Concerning the "quality" in terms of performance, the hardness (badness) of all the solutions is better (lower) than Sol_1. This behaviour is reasonable, since the more board are used, the closer to the optimal output of SWORDFISH are the sensors.

Table 19.3 Pareto solutions for Const=2

	Sol_2	Sol_3	Sol_4
$Board_0$	$\{s_0, s_5\}(0,0,0)$	$\{s_1, s_2, s_3\}(1,1,0)$	$\{s_4, s_6\}(2,2,0)$
$Board_1$	$\{s_0, s_5\}(0,0,0)$	$\{s_1, s_2,\}(1,1,0)$	$\{s_3, s_4, s_6\}(2,2,0)$
$Board_2$	$\{s_0, s_5\}(0,0,0)$	$\{s_1, s_2,\}(1,1,0)$	$\{s_4, s_6\}(2,2,0)$
$Board_3$			$\{s_3\}(2,1,0)$
Total cost	14.8	14.8	16.5
Hardness	78	78	61

The quantitative analysis produces a significant value added for the designer when the tradeoff is not so "obvious". In such a way, the driver may be not only the cost, but also the capability of the WSN to fulfil the initial application requirements.

19.10.2 Power Estimation of Selected Configuration

The modelling and simulation methodology illustrated in section 19.7 can be profitably used for evaluating power performance of WSN deployments so as to optimise parameter tuning or to design solutions aiming at improving standard operation. In situations where the positions of sensor nodes is carefully engineered before actual deployment - as e.g. in the precision agriculture experiment of Donnafugata where appropriate node positioning is essential for reaching sensing goals - the solution we

propose allows accurate evaluation of power consumptions in pre-deployment phase. This is essential for choosing the most suitable solution inside the design space of the foreseen application.

The simulator was used in a preliminary phase to evaluate, and possibly optimise power consumption for a WSN with application-specific requirements and topology such as the ones adopted in the Art-Deco case. Ten nodes send periodically measures (in packets of about 50 bytes) to a base station (the network topology is a star). From such general high level description of the application it is possible to design a model to simulate the data collection scenario (based on the IEEE 802.15.4 protocol).

Our investigation was on one side meant at identifying appropriate transmission periods (impacting on network duty cycle). Beaconed operation was chosen and a design space exploration on duty cycle of the network was performed[3]. The Beacon Order (BO) determines the temporal distance between two successive active parts, while the Superframe Order (SO) determines the length of such active parts. Therefore these two parameters together define the duty cycle of the network. Simulation were carried for 400 superframes[4].

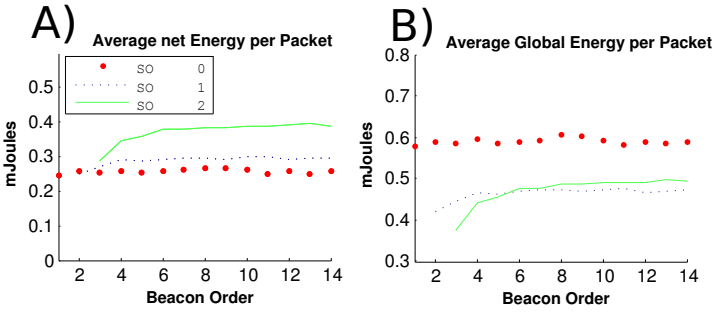

Fig. 19.5 Results of the simulation campaign, the energy was measured on an interval corresponding to 400 superframes, traffic was adapted to duty cycle of the network. The average net energy does not consider the contributions due to infrastructure communication but only the energy spent for transmitting packets (including possibly retransmissions).

In the graph A) of the figure 19.5 the average net energy that is necessary for transmitting a packet is shown, the graph suggests that power performance is better when SO is smaller. On the other hand graph B) representing the average global energy per packet (i.e. including the energy spent for infrastructure communication) shows that using SO 2 is optimal up to BO 5 then SO 1 is optimal. This can be explained considering the fact that if the superframe is longer more packets are sent in the same superframe and consequently the number of beacon messages that are received per application packet is smaller.

[3] It was established that the global throughput on the network was 50 kbps (5kbps per device) times the duty cycle, therefore by varying the duty cycle a different load was reached.

[4] The number of Superframes was chosen so as to guarantee reliable results. The temporal length depends on the BO parameter.

These results helped in devising that an adequate data transmission interval should be in the range of few minutes, considering the power requirements of the platform. Moreover they show that having an excessive inter-beacon distance tends to concentrate the transmissions at the beginning of the active part and therefore boosts the channel contentions and consequently power consumptions, therefore guiding towards the optimal choice of parameters.

19.11 Conclusions

In this Chapter we outlined the links existing between the application requirements and the constraints related to the environment in which the wireless sensor network will be deployed and the technology of the nodes. Particular attention has been paid to the task of modelling the *functional goal* of the sensor network and on deriving from that a set of guidelines and constraints to drive the following phases of node level and network level design and modelling for optimisation of energy related issues. It has been shown that the pure availability of a HW technology for the node and of some SW layer to support the distribution of applications it is not sufficient to provide a viable answer to basic questions regarding i) the feasibility of the project and ii) the fulfilment of severe design/application constraints. To cope with such needs, particular emphasis has been devoted to describe the main activities, actors and figures of merit involved in the different stages of a comprehensive design flow and on the crucial impact of the modelling of the node and network behaviour on the quality of the final results. The focus has been mainly kept at system level, while providing appropriate references to move more in depth into the covered topics.

References

1. Buck, J., Ha, S., Lee, E.A., Messerschmitt, D.G.: Ptolemy: a framework for simulating and prototyping heterogeneous systems, pp. 527–543 (2002)
2. Campanoni, S., Fornaciari, W.: Node-level optimization of wireless sensor networks, pp. 1– 4 (2008)
3. El-Hoiydi, A., Arm, C., Caseiro, R., Cserveny, S., alii: The ultra low-powerwisenet system. In: Proc. DATE 2006, vol. 1, pp. 1–6 (2006)
4. Fall, K., Varadhan, K.: The ns Manual (formerly ns Notes and Documentation). The VINT Project 16 (2006)
5. Harel, D.: Statecharts: A visual formulation for complex systems. Sci. Comput. Program., 231–274 (1987)
6. Heinzelman, W., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: System Sciences
7. Moser, C., Brunelli, D., Thiele, L., Benini, L.: Real-time scheduling for energy harvesting sensor nodes. Real-Time Syst. 37(3), 233–260 (2007)
8. Mura, M., Paolieri, M.: Sc2: State charts to system c: Automatic executable models generation. In: Proceedings FDL 2007, Barcelona, Spain (2007)

9. Mura, M., Paolieri, M., Fabbri, F., Negri, L., Sami, M.: Power modeling and power analysis of IEEE 802.15.4: a concurrent state machine approach. In: Proc. CCNC (2007)
10. Mura, M., Sami, M.G.: Code generation from statecharts: Simulation of wireless sensor networks. In: Euromicro Symposium on Digital Systems Design, pp. 525–532 (2008)
11. Negri, L., Chiarini, A.: StateC: a power modeling and simulation flow for communication protocols. In: Proc. FDL, Lausanne, Switzerland (2005)
12. Negri, L., Sami, M., Macii, D., Terranegra, A.: FSM–based power modeling of wireless protocols: the case of Bluetooth. In: Proc. ISLPED, pp. 369–374 (2004)
13. Sami, M., Sciuto, D., Silvano, C., Zaccaria, V.: An instruction-level energy model for embedded vliw architectures. IEEE Transactions on CAD 21(9), 998–1010 (2002)
14. Varga, A., Hornig, R.: An overview of the omnet++ simulation environment. In: Proc. Simutools (2008)
15. Zeng, X., Bagrodia, R., Gerla, M.: GloMoSim: a library for parallel simulation of large-scale wireless networks. ACM SIGSIM Simulation Digest 28(1), 154–161 (1998)