A Domain Specific Visual Language for Modeling Power-Aware Reliability in Wireless Sensor Networks (Extended version)

Javier Troya
Universidad de Málaga
Dept. Lenguajes y Ciencias de la Computación
Bulevar Louis Pasteu, 35, 29071
Málaga, Spain
javiertc@lcc.uma.es

Antonio Vallecillo
Universidad de Málaga
Dept. Lenguajes y Ciencias de la Computación
Bulevar Louis Pasteu, 35, 29071
Málaga, Spain
av@lcc.uma.es

ABSTRACT

Reliability is an attribute that appears in all quality models, so it is important to take it into account when developing any kind of system. Its evaluation at latter stages of the software development may force the re-engineering of important parts of the system, something very costly. This is why it should be raised to the system design phase. Among the systems where reliability is a crucial issue, some wireless sensor network (WSN) protocols aim to extend the networks lifetime as much as possible, so a more reliable network will live longer. Following a model-driven engineering (MDE) approach, we propose the use of domain specific visual languages (DSVLs) to model the reliability of systems based on components by means of in-place behavioral rules and by modeling how the state of the components changes. We have developed as well a DSVL for modeling and analyzing reliability properties of a WSN protocol based on local information, namely directional source-aware routing protocol (DSAP).

Categories and Subject Descriptors

C.2.2 [**Computer Communication Networks**]: Network Protocols; G.3 [**Probability and Statistics**]: Reliability and Life Testing

Keywords

Reliability, DSVLs, WSNs

1. INTRODUCTION

The modeling and analysis of non-functional properties is very important when developing different kinds of systems. Software quality assessment is often applied at system implementation time, which is normally too late because the problems arisen during implementation can force the re-engineering of important parts of the system, which is very costly. This is why it should be raised to the system design phase. In this paper we focus on a non-functional property of systems known as *reliability*, and we try to measure it at design time. Concretely, we analyze reliability when modeling systems based on components by means of domain specific visual languages (DSVLs), key parts in Model-Driven Engineering (MDE) for representing models and metamodels. In this way, we pursue the correct and complete specification of a system by including the specification and analysis of its reliability properties at design time. Specifically, in this work we will focus on system reliability in the context of wireless sensor networks (WSNs) and energy consumption.

In order to obtain reliability measures for a system, it is necessary to be clear about what reliability is and how it is defined. Different definitions have been given for it. For example, the *Wikipedia* defines it as "the ability of a system or component to perform its required functions under stated conditions for a specified period of time". In [8], Fenton and Pfleeger say that "the accepted view of reliability is the probability of successful operation during a given period of time". Musa provides in [16] a similar definition: "reliability for software products is the probability for the software to execute without failure for some specified time interval". A more general definition of reliability is given by the quality model in ISO 9126-1 [12]. There reliability is defined as "the capability of the software product to maintain a specified level of performance when used under specified conditions". One more definition, in this case for software reliability, is given by Wohlin et al. [26]: "the probability for failure-free operation of a program for a specified time under a specified set of operating conditions".

A WSN is made up of spatially distributed sensor nodes deployed over a certain area to monitor physical or environmental conditions, such as sound, pressure, temperature, vibration, humidity, and to cooperatively pass their data among the nodes. The range of applications of WSNs is large, and it includes military operations, habitat and environmental monitoring, area surveillance or remote sensing. Reliability is a crucial aspect in WSN applications, especially those deployed for real-time communication, since data delivery should be guaranteed. For this reason, it is very important the routing protocol chosen in each circumstance. There are many routing protocols already studied [10], which

can be classified in broad terms as fault-tolerant routing [6], geographic routing [9] and energy aware routing [21, 22]. In this work we focus in the last group, and concretely in the Directional Source Aware routing Protocol (DSAP) [13] in order to study its reliability, at design time, in the original implementation and some variants.

The DSAP and its variants aim to extend the life of the network as much as possible. In this sense, we are going to model and monitor reliability in terms of the network's lifetime. Thus, a protocol will be more reliable than another if it keeps the network working longer. This is in accordance with the definitions of reliability given by Fenton and Pfleeger [8] and Musa [16]. Throughout this paper, we will be presenting the DSAP and some variations for its routing. We model the DSAP and its reliability in the domain of MDE and in-place rules based on DSVLs. For the implementation and simulation, we use *e-Motions* [18], a graphical framework and tool for defining timed behavioral specifications of models. We show as well how different kinds of reliability analysis can be carried out and perform experiments based on realistic situations.

After this introduction, in Section 2 we present an approach for modeling reliability of systems based on components in design phases. In Section 3 we present a DSVL to model the DSAP, some variants of it, and to monitor and analyze its reliability. Sections 4 and 5 present some related work and conclusions, respectively.

2. MODELING RELIABILITY IN DSVLS

2.1 Modeling Behavior

As presented in [25], we specify the dynamic behavior of a DSVL by describing the evolution of the modeled artifacts along some time model. Following an MDE approach, we achieve this by applying model transformations supporting in-place update. In this way, a system starts executing in a particular state, and it evolves over time by the non-deterministic firing of the behavioral rules.

Semantics are precisely specified by a set of behavioral rules, each of which represents a possible *action* of the system. These rules are of the form $l : [\text{NAC}]^* \times \text{LHS} \rightarrow \text{RHS}$, where l is the rule's label (its name); and LHS (left-hand side), RHS (right-hand side), and NAC (negative application conditions) are model patterns that represent certain (sub-) states of the system. The LHS and NAC patterns express the precondition for the rule to be applied, whereas the RHS one represents its postcondition, i.e., the effect of the corresponding action. Thus, a rule can be applied, i.e., triggered, if an occurrence (or match) of the LHS is found in the model and none of its NAC patterns occurs. Generally, if several matches are found, one of them is non-deterministically selected and applied, producing a new model where the match is substituted by the appropriate instantiation of its RHS pattern (the rule's *realization*). The model transformation proceeds by applying the rules in a non-deterministic order, until none is applicable.

2.2 Probabilistic Rules

In many kinds of systems, state machines are good for defining the state of their components. If these state machines only have two states, the transition from one state always brings to the other one. However, in the case of components having more than one possible state, their state machines can transit from one state to more than one other state. Our in-place rules are triggered when there is a match of their LHS in the system. We want to model that the triggering of a rule can make the system evolve to more than one possible state, as happens in state machines, according to a given probability for each transition. This is specially useful when we are interested in modeling the reliability of systems.

Let us first introduce the concepts of *mean time to failure* (MTTF) and *mean time to repair* (MTTR). The former represents the average from the time that a component of the system is put into service until it first experiences failure, while the latter is the average time that the component is out of service before it is repaired. As explained in [17], both times are normally modeled with exponential distributions. We already presented in [24] how we are able to apply many probabilistic distributions to our in-place rules' internal variables and durations. In this way, to model the MTTF of every component in our system, we only need a very simple rule where there is a component in its LHS which changes its state in the RHS. The duration of the rule will follow an exponential distribution whose parameter is the mean time to failure of the component. Figure 1 shows this rule in a system where the components are chips. A similar rule would be needed for modeling the MTTR.

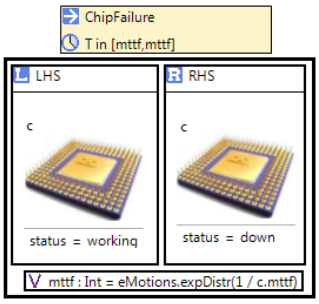


Figure 1: ChipFailure Rule

Since we want to find a way of modeling systems whose components can have more than two states, we introduce in-place rules with more than one RHS. Imagine we have a system made up of chips, where each one can be in one of three states: fully working (fw), partially working (pw) or down (d). When a chip is in any of the three states, it can transit to one of the other two with a given probability (see Figure 2 for an example).

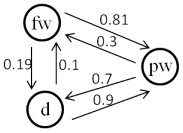


Figure 2: Transitions and probabilities

Now, to model the failure of a chip, we have to consider that it can evolve from the fully working state to either partially working or down states. Thus, the rule modeling this has

two RHSs, and it evolves to one of them according to a probability. As before, the duration of the rule is exponentially distributed. Consequently, the rule is "twice" probabilistic: (1) due to the probability of transitions and (2) to the exponential duration. Such rule is shown in Figure 3. It is very simple, and since the only difference among its two RHSs is the value of the status attribute of the chip, it could be modeled with only a RHS and an OCL *if* condition in that attribute. However, this same approach can be used to create rules whose several RHSs differ completely.

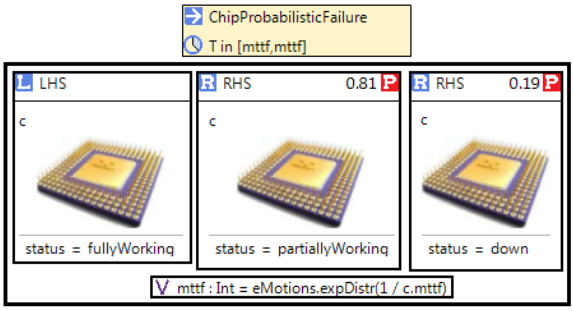


Figure 3: Rule with two RHSs

2.3 Observers

In [25] we presented the use of *observers* in order to specify and calculate the value of non-functional properties of systems. An observer is an object whose purpose is to monitor the execution of the system: the state of the objects, of the actions, or both. Observers are defined by means of a metamodel which is then merged together with the system metamodel in a non-intrusive way. This allows to introduce observers in the behavioral rules. Many non-functional properties can be specified and monitored, such as throughput, delays, cycle times, busy times, etc. In this work we focus on reliability in terms of energy consumption in system components. We go into this in next section.

3. RELIABILITY IN WSNS

In this section we apply our approach for specifying and measuring the reliability of systems modeled with DSVLs. Concretely, we model the structure and behavior of a routing protocol for WSNs named DSAP.

3.1 The DSAP

The Directional Source Aware routing Protocol (DSAP) [20] was designed for low-power fixed wireless topologies and based on local information where each node only knows about its neighbors information. It has several advantages over other routing protocols, including incorporating power considerations and having no routing table [20]. Each node has a unique ID, which gives how far, in terms of number of nodes, the node is from the network perimeter in each direction. For example, the ID of the node numbered 43 in the WSN in Figure 4 is $(3, 3, 4, 4, 6, 5, 5, 3)$. This means that there are three nodes (42, 41, 40) to the edge in direction 0 (left), three nodes (32, 21, 10) in direction 1 (up-left), etc. Consequently, the ID is a vector with as many components as neighbors have the nodes in the network – 8 in our case.

When transmitting a packet, each node contains information about its neighbors' IDs and the ID of the packet's

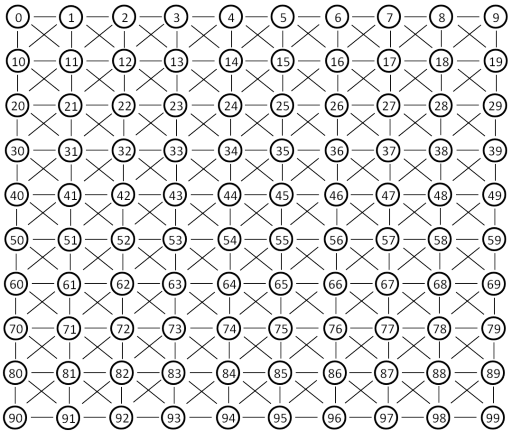


Figure 4: 100 nodes WSN net

target node. In order to choose which neighbor a packet is forwarded to, the Directional Value (DV) is used. The DV of each neighbor is calculated by taking their IDs and subtracting them from the destination node's ID. Let us imagine that node 22 wants to send a packet to node 77. The original DSAP can be seen as a two-step routing protocol.

First, the source's and target's IDs are subtracted. The result in our case is $(2, 2, 2, 2, 7, 7, 7, 2) - (7, 7, 7, 2, 2, 2, 2, 2) = (-5, -5, -5, 0, 5, 5, 5, 0)$. This obtained vector indicates which neighbors the packet can be forwarded to: those with a non-positive number are discarded. In this way, the packet will not be sent to nodes in directions 0 (left – 21), 1 (up-left – 11), 2 (up – 12) and 3 (up-right – 13). Second, the ID of those candidate neighbors to receive the packet is subtracted from the destination node's ID. The absolute values of the components in each resulting vector are added, which gives us the DV, and the neighbor with the smallest result is chosen. If there are more than one with the same value, one of them is randomly selected. In our case, the DVs for the nodes numbered 23, 33, 32 and 31 are, respectively, 28, 26, 28 and 32. As it was obvious, node 33 is closer to the destination and is the one chosen.

Variants. There have been some new routing methods proposed for improving the DSAP in order to extend the sensor network lifetime, such as the power aware routing [19]. However, in every proposed routing, the neighbor node which has the most power and shortest path is chosen most of the time. This causes the energy in the same nodes to be depleted, and creates an unbalanced power dissipation in the network. Besides, since the protocol always tries to forward the packet to a neighbor closer to the destination, some of the nodes in the network will stay untouched, whereas they could be chosen as an alternative path to prolong the overall network lifetime.

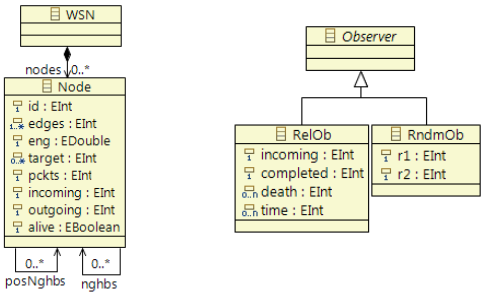
In this paper we select some protocols already proposed and consider a variant of the power-aware DSAP. Furthermore, we consider all possible directions from a given node, even if in the original DSAP the substraction of the source node's ID and target node's ID gives a negative number in a given direction. In this way we try to consider all possible paths and extend the network's lifetime as much as possible.

3.2 Modeling the DSAP and its Variants

In this section we present how to model the DSAP and its variants and their reliability in terms of energy consumption, and explain how they could be extended and made more real with the approach presented in Section 2. We follow an MDE approach and propose the specification of a DSVL for the high-level modeling and analysis of the protocols, in terms of behavioral rules, in the design phases. We show how once we model the original DSAP, modeling each of its variants is trivial.

3.2.1 Defining the Metamodels

The first step is to define a metamodel for the DSAP in WSNs, which describes their static structure. Our proposed metamodel is the simple one shown in Figure 5(a). According to it, a WSN is composed of a set of Nodes. Each Node has an identifier given by an integer (id – 0, 1, 2 and so on). Another identifier, which gives the position of the node in the network according to what was explained in Section 3.1, is kept in the edges attribute. It is a sequence with n components in a n-neighbors network. The remaining energy is kept in eng. The attribute named target contains the edges of the target(s) node(s) and pckts contains the number of packets that a node currently contains. These two attributes work like this: if there are no packets in a node, target is an empty sequence; if there is one packet, target contains a sequence with n components (in a n-neighbors network); if there are more than one packet, target contains a vector with $n * pckts$ components, where the first n components represent the edges of the target node of the packet that arrived first, and so on. Attributes incoming and outgoing contain the total number of incoming and outgoing packets in the node, and alive is a boolean stating if the energy of the node is above 0 (alive=true) or not (alive=false). If a node is not alive, it can neither receive messages nor forward them. As for the references, every node has a link (nghbs) to each neighbor and another link only to the neighbors which are alive (posNghbs).



(a) WSN Metamodel. (b) Obs Metamodel.

Figure 5: Metamodels

With the metamodel described we can already define behavioral rules for the functional behavior of the DSAP. Nevertheless, let us first introduce the Observers metamodel in order to be able to introduce observers in the rules and monitor reliability properties. It is presented in Figure 5(b). There are two observers, named RelOb and RndmOb. The former keeps the number of packets that arrive to the network (incoming) and those that reach their target node (completed). It also contains two sequences, death and time, that store the identifier of the nodes which die and the time they die. The latter observer is used to randomly select nodes from and to which packets are sent.

3.2.2 Defining the Behavior

Here we define the behavior of the DSAP for 8-neighbors WSNs as that presented in Figure 4. Modeling the behavior for a n-neighbors WSNs is trivial. We want to carry out two different analysis, one modeling the reality and another one for a more specific study. In the former, packets arrive at nodes randomly selected and their target node is also random. In the latter, packets are always sent from node 22 to node 77 (Figure 4).

To take into account the energy of the nodes and the size of the packets, we use a simple radio model described in [13]. In such model, the radio dissipates $E_{elec} = 50nJ/bit$ to run the transmitter or receiver circuitry and $E_{amp} = 100pJ/bit/m^2$ for the transmit amplifier to achieve an acceptable E_b/N_0 (see Figure 6 and Table 1) [11]. To transmit a k-bit message a distance d meters using this radio model, the radio expends $E_{Tx}(k,d) = E_{Tx-elec}(k) + E_{Tx-amp}(k,d) = E_{elec} * k + E_{amp} * k * d^2$. To receive this message, the radio expends: $E_{Rx}(k) = E_{Rx-elec}(k) = E_{elec} * k$.

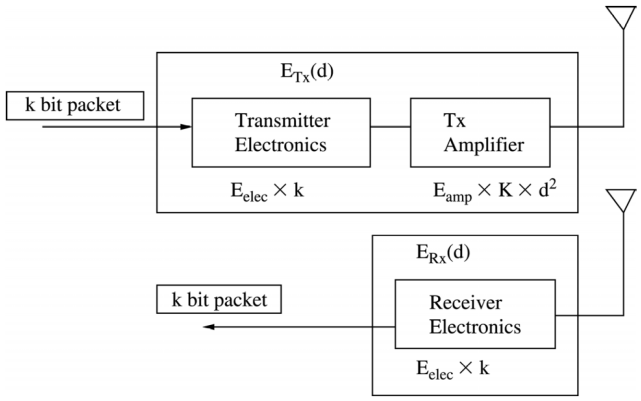


Figure 6: First-order radio model (from [19])

Table 1: Radio Characteristic (from [11])

Operation	Energy Dissipated
Transmitter Electronics ($E_{Tx-elec}$) Receiver Electronics ($E_{Rx-elec}$) ($E_{Tx-elec} = E_{Rx-elec} = E_{elec}$)	$50nJ/bit$
Transmit Amplifier (E_{amp})	$100pJ/bit/m^2$

We assume as well that the distance between the wireless nodes is equal to each other and all data packets contain the same number of bits. The parameters are the following: distance $d = 0.5m$ and number of bits transmitted $k = 512bits$. In short, every node dissipates 25612.8 nJ in the transmission of a packet and 25600 nJ in its reception. We suppose that every packet starts with an energy of 1000000 nJ.

Our initial model is the network composed of 100 nodes shown in Figure 4. There are also a RelOb and a RndmOb

observers. We have a rule named CalculateRandom where it is decided the source and target nodes for the next incoming packet. Their ids are kept in the observer's attributes r1 and r2. Rule PacketArrival (Figure 7) uses the values in such attributes to model the arrival of the packet. It arrives to node n0, while its target is node n1 (LHS). Notice the corresponding update of the necessary attributes in the rule's RHS.

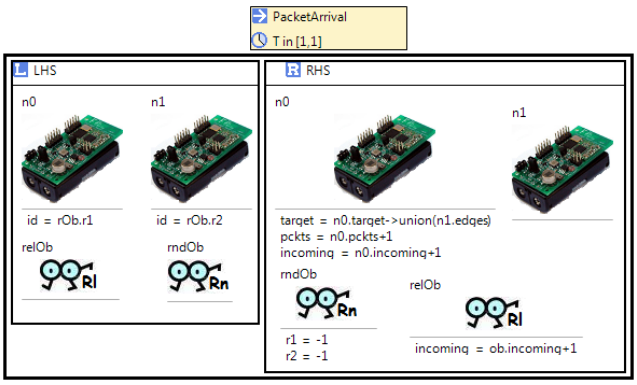


Figure 7: PacketArrival Rule

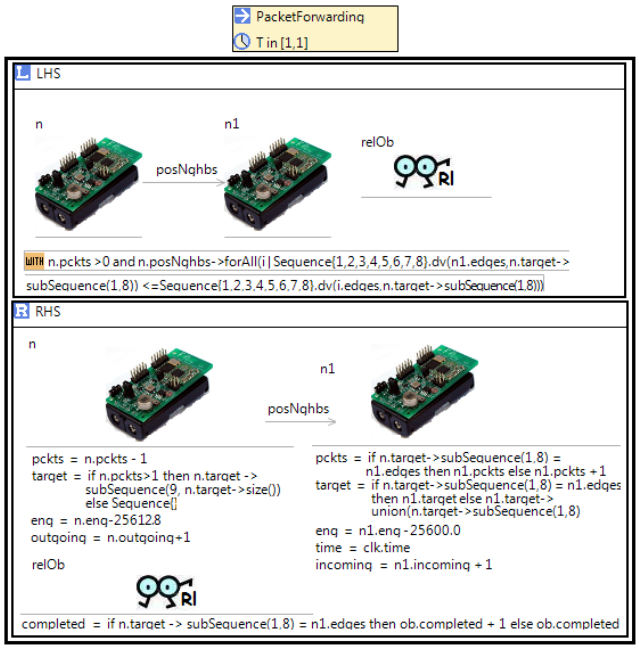


Figure 8: PacketForwarding Rule

Rule PacketForwarding (Figure 8) models the forwarding of a packet to the alive neighbor with the lowest DV. The OCL condition in the LHS checks that node n0 has a packet and n1 is a neighbor with positive energy and the lowest DV. That expression uses a helper, named dv, which deals with the calculation of the DV and is: *context Sequence::dv(s1 : Sequence, s2 : Sequence): Integer body: self -> iterate(i ; acc : Sequence = Sequence{} | acc->append(i.abs()))->sum()*. In the rule's RHS, the attributes of the nodes are modified with the updating of the energy, the number of packets contained, incoming and outgoing packets, etc. The RelOb observer updates its completed attribute when n1 is the target node of the packet being forwarded.

To model some of the variants of the DSAP, we only need to change the OCL condition in the PacketForwarding rule's LHS. For example, we can model the Power-DSAP with power-aware routing [20]. It selects the paths according to the ratio of the directional value and the power available at the neighboring nodes. The OCL condition would be this: *n.pckts > 0 and n.posNghbs -> forAll(i | Sequence{1,2,3,4,5,6,7,8}.dv(n1.edges, n.target -> subSequence (1,8)) / n1.eng <= Sequence{1,2,3,4,5,6,7,8}.dv(i.edges, n.target -> subSequence(1,8)) / i.eng)*. We have developed another variant, where the routing selects the path according to the ratio of the DV, the power remaining at the neighboring nodes and it also takes into account the packets contained at the neighboring nodes and the power they will spend in forwarding them. The OCL condition of the PacketForwarding rule's LHS would be the following: *n.pckts > 0 and n.posNghbs -> forAll(i | Sequence{1,2,3,4,5,6,7,8}.dv(n1.edges, n.target -> subSequence (1,8)) / (n1.eng - n1.pckts* 25612.8) <= Sequence{1,2,3,4,5,6,7,8}.dv(i.edges, n.target -> subSequence(1,8)) / (i.eng - i.pckts*25612.8))*.

Regarding the rules' duration, notice that we have established that a new packet arrives every time unit and nodes forward packets (as long as they have any) every time unit too. To make the model more realistic, these times should have followed some probabilistic distribution, something that we are able to model as we already presented in [24]. However, to make the simulations the least random possible, we decided to set these times as 1 in order to compare the different protocols in the different simulations in a fair way.

The complete specification of the DSAP can be consulted at [2]. It contains for example a rule to keep in the RelOb a list with the death nodes and the time they die.

Extensions to the DSAP. In order to make the modeling and simulations more realistic and according to real-life situations, we can add a few rules for that purpose. In real-life WSNs, nodes consume energy when they are in stand-by. The failure of nodes can also be modeled, with rules similar to those in Figures 1 and 3. We have not had this into account in the workshop paper due to space limitations, nor have the works presented in [13, 20, 19]. However, in this extended version we run experiments where we model power consumption of nodes in stand-by and also the death of nodes due to other circumstances. Likewise, we can model the repair of nodes which run out of energy or simply break.

3.3 Reliability Analysis

Once we have the specifications of the protocols modeled in *e-Motions*, we are able to simulate them and analyze their results. These specifications are automatically transformed to a domain with well-defined semantics, namely Real-Time Maude [**?**], by means of ATL [**?**] transformations. We show results with a 100-nodes and 8-neighbors network, such as the one evaluated in [19], and with a 12-nodes and 4-neighbors network, like the case study used in [13].

3.3.1 100-nodes and 8-neighbors network

As we mentioned in Section 3.2.2, we want to carry out two different analysis. In one of them, the source and target nodes of packets are chosen randomly, while in the other packets always go from node 22 to node 77 (Figure 4). The first one models more accurately the reality, while the second one allows us to focus on nodes 22 and 77 and their neighbours. For both kinds of analyses, we have run three protocols: (1) the original DSAP [13], (2) the power-aware DSAP [19] and (3) a new variant that also takes into account the packets that still need to be processed in each node apart from the remaining energy. Let us call the last one power-aware DSAP v2. Recall that in the three protocols we consider all the neighboring nodes as candidate nodes to forward a packet and not only those whose subtraction with the target node is positive.

Random Arrivals

In these simulations, the RndmOb observer deals with the random selection of the source and target nodes for every packet. We are interested in knowing the lifetime of the network in each protocol. For that, we consider that the network dies when a node consumes all its energy. So we will measure the reliability in terms of the time the network has been alive and the number of packets completed (those that reached their destination) within that time.

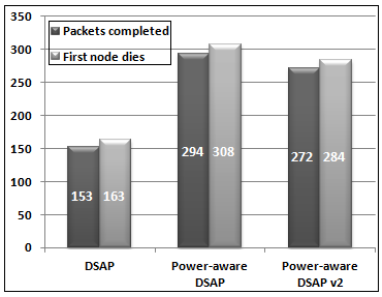


Figure 9: Results with random arrivals

As we already knew from other works [20, 19], and as we can see in the simulation results shown in Figure 9, the power-aware DSAP is more reliable than the original DSAP. Besides, it turns out that it is also more reliable than the new developed protocol that also takes into account the packets that still need to be processed in the nodes.

Fixed Arrivals

We have made these simulations stop when all the energy in the nodes around 22 or 77 is consumed, so that no packet can reach its destination (77). We have made node 22 not to consume any energy when forwarding packets and node 77 when it receives packets. Otherwise, the energy in these nodes would be consumed very soon.

In Figure 10 we can see the time at which the energy in each node around node 22 is consumed for each protocol. Let us focus first in the original DSAP. Since it does not take into account the remaining energy in the nodes, it always follows the shortest path to the destination. This is why it always uses node 33 from 22 to send packets to node 77

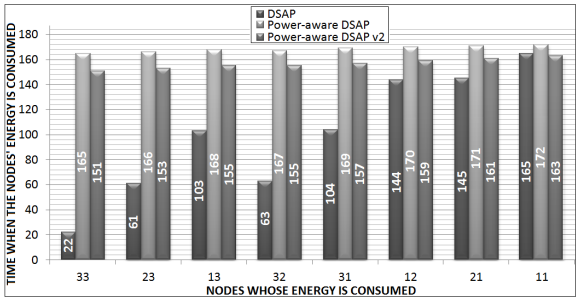


Figure 10: Results with fixed arrivals

at the beginning. This makes the energy in node 33 to be consumed quickly, in time 22. Then, the protocol forwards packets to nodes 23 and 32 from 22 because they are at the same distance from the destination. The energy at both nodes is consumed almost at the same time, at times 61 and 63, respectively. The same happens then with nodes 13 and 31, and later with nodes 12 and 21. The last node to run out of energy is the 11. The number of packets that reach node 77 are 87.

In the other two protocols, the energy consumption in the nodes is much more uniform, since they take into account the remaining energy at nodes. In the power-aware DSAP, the energy consumption is slightly more uniform than in the power-aware DSAP v2, and the packets completed are 128, while in the latter protocol they are 123.

From the three analyzed protocols, the power-aware DSAP already presented in [20, 19] is the most reliable both with random and fixed arrivals.

3.3.2 12-nodes and 4-neighbors network

In this section we make experiments with the 12-nodes and 4-neighbors network shown in Figure 11. These experiments are more realistic since we consider power consumption of nodes in stand-by and failure of nodes due to other circumstances apart from energy depletion. In fact, WSNs can be deployed in many and different environments, so there are different causes that can make nodes die. For example, animals could try to eat them, or they could be damaged due to environmental conditions.

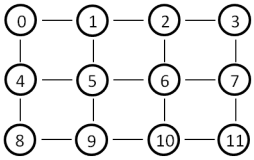


Figure 11: 12 nodes WSN net

The chart in Figure 12 displays three different executions in such network:

- The blue one (*Power-DSAP*) is an execution with the Power-DSAP [13]. Recall that every node dissipates 25612.8 nJ in the transmission of a packet and 25600 nJ in its reception.

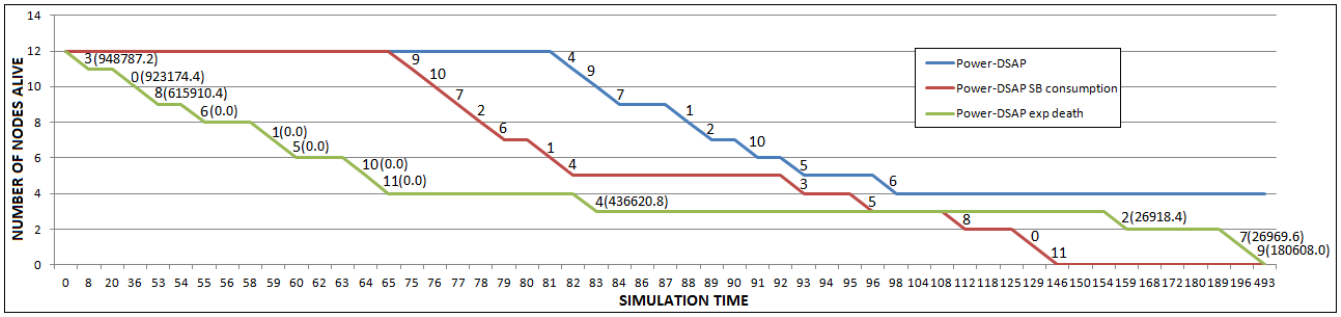


Figure 12: Death of nodes along time

- The red one (*Power-DSAP SB consumption*) uses the Power-DSAP and it also considers the power consumption of nodes in stand-by. Concretely, every node consumes 3000 nJ every unit of time. The rule used to model this is the simple one shown in Fig. 13.

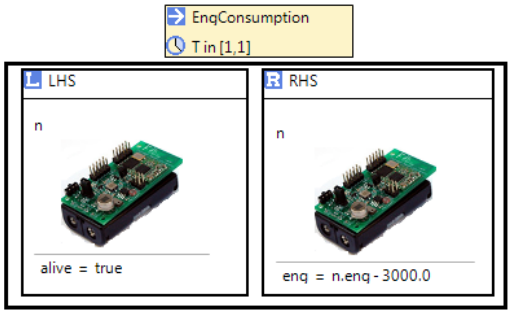


Figure 13: Power consumption in stand-by

- The green one (*Power-DSAP exp death*) applies the Power-DSAP and it also considers the failure in nodes due to other circumstances than the energy depletion. Such failure follows an exponential distribution of mean 200 units of time. This concept of "random" failure or death is the same as the *mean time to failure* presented in Section 2.2. To model such death, the rule shown in Fig. 14 has been used, which follows the pattern presented in Fig. 1. Note that the RelOb observer has a new attribute, eng, apart from those shown in Fig. 5(b). It is of type Sequence and we use it to store the remaining energy in nodes when they die.

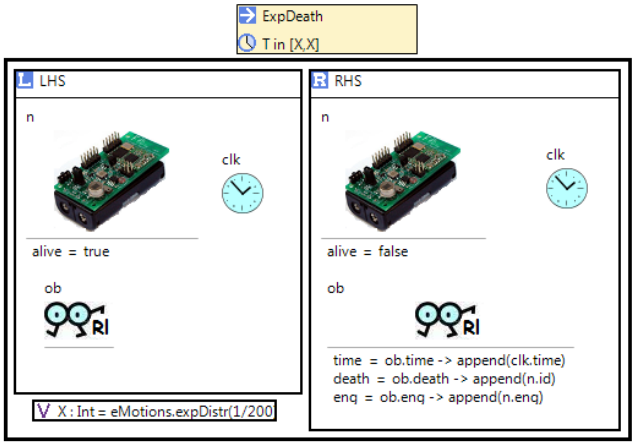


Figure 14: Death due to random circumstances

The X axis of the chart shows the simulation time, while the Y axis indicates the number of nodes alive. It can be seen as well the concrete nodes that die (indicated by the numbers on the lines). In the case of the *Power-DSAP exp death*, it is indicated between brackets the energy available in the nodes when they die. In the three cases, packets arrive at random nodes, and their destination is also random. Let us analyze each of the three cases.

In the Power-DSAP, the first node (number 4) dies due to energy depletion in time 82. From that moment, several nodes die, until node number 6 does in time 98. From that moment, the nodes alive are 0, 3, 8 and 11, which cannot exchange packets among them, so there is not packet flow any longer. That is the reason why no more nodes die from time 98. The number of packets that arrive to their destination is 83.

In the Power-DSAP with power consumption of nodes in stand-by, the first node (number 9) dies due to energy depletion in time 71. It happens earlier in simulation time than in the previous case due to energy consumption in stand-by. From that moment, several nodes (10, 7, 2, 6, 1 and 4) run out of energy until time 82. At that point, only nodes 0, 3, 5, 8 and 11 are alive. They cannot exchange packets between them because they are not directly connected. This is why they die later in time: the remaining energy in those nodes from unit of time 82 is depleted only due to power consumption in stand-by, being no packet flow any longer. The last node to die is number 11 in time 146. 75 packets arrive to their destination in this case.

In the Power-DSAP that considers the death of nodes due to other circumstances apart from energy depletion, the first node (number 3) dies in time 8. When it dies, its energy is 948787.2 nJ, so it dies due to a random circumstance. The same happens with nodes 0 and 8, which also die quite early in time. Having three nodes less already in time 53, the remaining nodes have to work more. This causes the energy of node 6 to be depleted in time 55, and the energy in nodes 1, 5, 10 and 11 to be depleted in times 59, 60, 64 and 65, respectively. From unit of time 65, the only nodes alive are

2, 4, 7 and 9. Their energy is not consumed anymore since they cannot exchange packets, so all of them die due to other circumstances. The last one to do it, is node 9 in time 493. The number of packets that arrive to their destination is 51.

With these experiments we have shown that we are able to model and simulate real-life scenarios that were not considered in the papers that dealt with the DSAP [13, 20, 19]. The values chosen for quantity of energy consumption per time unit and the distribution followed for the failure of nodes due to random circumstances could have been different depending on the network. In any case, our experiments serve as proof of concept for our approach.

4. RELATED WORK

As we do, the work in [15] presents by means of an MDE approach a DSVL to model WSNs. According to their approach, the WSN models described are then to be translated to WSNs domain specific textual languages by means of model-to-model and model-to-text transformations, in order to later simulate them. In this way, they only describe the static structure for WSNs, but not its dynamics. In fact, they add behavioral elements in the static models. In our approach we include the simulation and reliability analysis of the networks. As far as we are concerned, our work is the first that proposes an approach to model and simulate WSNs using high-level DSVLs. Furthermore, no other work has previously presented the use of several RHSs based on probabilities in transformation rules, which allows us to model systems in a more realistic way.

The works in [13, 20, 19] present and describe the DSAP and also compare some of its variants in terms of network lifetime. However, in none of these works it is mentioned the way the protocols are implemented, although we believe it is in a lower level than ours, nor the platform or program used to simulate them. They have not taken into account either real-life scenarios where power consumption in stand-by or death of nodes due to random circumstances are considered. Some other works have also studied the reliability in WSNs, such as the one presented in [1], where they do consider the failure probability of each sensor. Their algorithm considers different types of topologies for the network, while the DSAP and variants only consider local information, and is implemented using dynamic programming to compute reliability. The RPAR protocol [4] is also a power-aware routing protocol for WSNs that tries to maximize the network lifetime by dynamically adapting packets forwarding.

Regarding the way we have considered reliability in WSNs in this work, there are similarities and differences when compared to other works. For example, the authors in [1] define reliability of a WSN cluster as the probability that "a minimum aggregate rate of information can be delivered to the sink node". In many works [14, 3, 7], reliability in WSNs has to do with link reliability, this is, reliability between sensor nodes. Thus, probabilities of successfully sending/receiving packets for every pair of connected nodes have to be given. From such probabilities, the overall network reliability is calculated. In [5], the authors relate reliability to the percentage of data expected to arrive at the sink nodes. They also consider that nodes may run out of energy, may be damaged or stolen, as we do. The authors in [23] relate reliability to the success of every individual packet transmission. According to them, such reliability depends on the distance and the power dissipation in transmissions. Consequently, they try to minimize the power consumption, as our implemented protocol does. In other works, the authors speak about reliability in WSNs but do not give a concrete definition for it. As an example, the authors in [4] do not explicitly state the way they define reliability, although the reader can deduce that it is related to the number of deadlines missed and energy consumption.

5. CONCLUSIONS

In this paper we have presented an MDE approach to perform a high-level modeling of the reliability in systems based on components by describing how the state of their components can vary. We have introduced the use of several RHSs in transformation rules for describing the behavior of components in terms of their reliability. We have also presented how we can easily model the DSAP for WSNs and some variants by simply realizing small changes in the rules. By using observers we have been able to monitor and analyze the reliability of the protocols in terms of energy consumption in nodes. Furthermore, we have described how we extend the modeling of the protocols by including some behavioral rules that allow more precise and realistic simulations by modeling energy consumption of nodes in stand-by or how nodes fail due to random circumstances.

Acknowledgments. This work has been supported by Spanish Research Project TIN2011-23795.

6. REFERENCES

[1] H. AboElFotoh, E. ElMallah, and H. Hassanein. On The Reliability of Wireless Sensor Networks. In *ICC '06*, volume 8, pages 3455 –3460, june 2006.

[2] Atenea. DSAP and its Reliability in *e-Motions*, 2012. `http://atenea.lcc.uma.es/index.php/Main_Page/Resources/E-motions/DSAP`.

[3] E. Cañete, M. Díaz, L. Llopis, and B. Rubio. Hero: A hierarchical, efficient and reliable routing protocol for wireless sensor and actor networks. *Computer Communications*, 35(11):1392 – 1409, 2012.

[4] O. Chipara, Z. He, G. Xing, Q. Chen, X. Wang, C. Lu, J. Stankovic, and T. Abdelzaher. Real-time Power-Aware Routing in Sensor Networks. In *IWQoS '06*, pages 83 –92, june 2006.

[5] A. Czubak and J. Wojtanowski. Lifespan-aware routing for wireless sensor networks. In *Proceedings of the 4th KES international conference on Agent and multi-agent systems: technologies and applications, Part II*, KES-AMSTA'10, pages 72–81, Berlin, Heidelberg, 2010. Springer-Verlag.

[6] A. Datta. Fault-Tolerant and Energy-Efficient Permutation Routing Protocol for Wireless Networks.

[7] P. P. Ed, P. R. Pereira, A. Grilo, F. Rocha, M. S. Nunes, A. Casaca, C. Chaudet, P. Almström, and M. Johansson. END-TO-END RELIABILITY IN WIRELESS SENSOR NETWORKS: SURVEY AND RESEARCH CHALLENGES, Dec. 2007.

[8] N. E. Fenton and S. L. Pfleeger. *Software Metrics: A Rigorous and Practical Approach*. PWS Publishing Co., Boston, MA, USA, 1998.

[9] H. Frey and I. Stojmenović. *Geographic and energy-aware routing in sensor networks*, chapter 12. Wiley, 2005.

[10] D. Goyal and M. R. Tripathy. Routing Protocols in Wireless Sensor Networks: A Survey. *ACCT12*, 0:474–480, 2012.

[11] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8 - Volume 8*, HICSS '00, pages 8020–, Washington, DC, USA, 2000. IEEE Computer Society.

[12] ISO/IEC. *ISO/IEC 9126. Software engineering – Product quality – Part 1: Quality model.* ISO/IEC, 2001.

[13] K. Jalil and M. Nategh. A Composed Energy Aware Metric for WSNs. In *ICCDA*, volume 2, june 2010.

[14] Y. Li, C. S. Chen, Y.-Q. Song, and Z. Wang. Real-time QoS support in wireless sensor networks: a survey. In *Proc. 7th IFAC Int Conf on Fieldbuses & Networks in Industrial & Embedded Systems (FeT'07)*, 2007. Survey.

[15] F. Losilla, C. Vicente-Chicote, B. Álvarez, A. Iborra, and P. Sánchez. Wireless Sensor Network Application Development: An Architecture-Centric MDE Approach. In F. Oquendo, editor, *ECSA*, volume 4758 of *LNCS*, pages 179–194. Springer, 2007.

[16] J. D. Musa. *Software Reliability Engineering.* McGraw-Hill Inc., New York, NY, USA, 1998.

[17] L. B. Page. *Probability for Engineering With Applications to Reliability.* Computer Science Press, Inc., USA, 1989.

[18] J. E. Rivera, F. Durán, and A. Vallecillo. A graphical approach for modeling time-dependent behavior of DSLs. In *Proc. of VL/HCC'09*, Oregon (US), 2009.

[19] A. Salhieh and L. Schwiebert. Power-Aware Metrics for Wireless Sensor Networks. *International Journal of Computers and Applications*, 26(4), 2004.

[20] A. Salhieh, J. Weinmann, M. Kochhal, and L. Schwiebert. Power Efficient Topologies for Wireless Sensor Networks. In *International Conference on Parallel Processing*, pages 156–163, 2001.

[21] R. C. Shah and J. M. Rabaey. Energy Aware Routing for Low Energy Ad Hoc Sensor Networks. In *WCNC'12*, March 2002.

[22] Y. Tian, E. Ekici, and F. Özgüner. Energy-Constrained Task Mapping and Scheduling in Wireless Sensor Networks. In *MASS*. IEEE, 2005.

[23] L. Tran-Thanh and J. Levendovszky. A novel reliability based routing protocol for power aware communications in wireless sensor networks. In *Proceedings of the 2009 IEEE conference on Wireless Communications & Networking Conference*, WCNC'09, pages 2308–2313, Piscataway, NJ, USA, 2009. IEEE Press.

[24] J. Troya and A. Vallecillo. A Domain-Specific Language to Specify and Simulate Queuing Network Models. Submitted, Dec. 2011.

[25] J. Troya, A. Vallecillo, F. Durán, and S. Zschaler. Model-Driven Performance Analysis of Rule-Based Domain Specific Visual Models. *Information and Software Technology, available online*, 2012. `http://www.sciencedirect.com/science/article/pii/S0950584912001358`.

[26] C. Wohlin, P. Höst, P. Runeson, and A. Wesslén. *Software Reliability*, volume 15. Encyclopedia of Physical Sciences and Technology (third edition), 2001.