Model-Based Validation of QoS Properties of Biomedical Sensor Networks

Simon Tschirner
Uppsala University
P.O. Box 337
751 05 Uppsala, Sweden
simon.tschirner@it.uu.se

Liang Xuedong
Rikshospitalet University
Hospital and University of Oslo
P.O. Box 1139
N-0316 Oslo, Norway
xuedongl@medisin.uio.no

Wang Yi
Uppsala University
P.O. Box 337
751 05 Uppsala, Sweden
yi@it.uu.se

ABSTRACT

A Biomedical Sensor Network (BSN) is a small-size sensor network for medical applications, that may contain tens of sensor nodes. In this paper, we present a formal model for BSNs using timed automata, where the sensor nodes communicate using the Chipcon CC2420 transceiver (developed by Texas Instruments) according to the IEEE 802.15.4 standard. Based on the model, we have used UPPAAL to validate and tune the temporal configuration parameters of a BSN in order to meet desired QoS requirements on network connectivity, packet delivery ratio and end-to-end delay. The network studied allows dynamic reconfigurations of the network topology due to the temporally switching of sensor nodes to power-down mode for energy-saving or their physical movements. Both the simulator and model-checker of UPPAAL are used to analyze the average-case and worst-case behaviors. To enhance the scalability of the tool, we have implemented a (new text-based) version of the UPPAAL simulator optimized for exploring symbolic traces of automata containing large data structures such as matrices. Our experiments show that even though the main feature of the tool is model checking, it is also a promising and competitive tool for efficient simulation and parameter tuning. The simulator scales well; it can easily handle up to 50 nodes in our experiments. The model checker installed on a notebook can also deal with networks with 5 up to 16 nodes within minutes depending on the properties checked; these are BSNs of reasonable size for medical applications. Finally, to study the accuracy of our model and analysis results, we compare simulation results by UPPAAL for two medical scenarios with traditional simulation techniques using OMNeT++, one of the most used simulation tools for wireless sensor networks. The comparison shows that our analysis results coincide with the simulation results by OMNeT++ in most cases although there are some differences caused the simplified wireless channel model in UPPAAL.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
EMSOFT'08, October 19–24, 2008, Atlanta, Georgia, USA.
Copyright 2008 ACM 978-1-60558-468-3/08/10 ...$5.00.

Categories and Subject Descriptors

D.2.2 [**Software**]: Software Engineering—*Design Tools and Techniques*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms

Design, Performance, Verification

Keywords

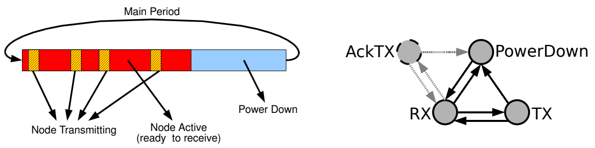
Modelling and Verification, Wireless Sensor Networks

1. INTRODUCTION

Wireless Sensor Networks (WSN) [1] contain hundreds or thousands of sensor nodes equipped with sensing, computing and communication devices. These sensor nodes may be distributed in a large area and connected by short-range communication devices over wireless channels. WSNs have a lot of potential applications, e.g., battlefield surveillance, wildlife monitoring and medical applications. In these mission-critical applications, a certain set of QoS requirements on network performance must be satisfied. This poses a number of challenges on the design and analysis of WSNs. Due to the severe constraints on hardware platform, dynamic working environments, and self-organizing manner, a key design challenge is to evaluate the network performance without investing on the hardware platforms and the time-consuming deployment and measurement.

In this paper, we demonstrate that model-based techniques can be used as an alternative approach to the design and analysis of WSNs to complement traditional simulation-based techniques. We shall study Biomedical Sensor Networks (BSN), which are small-size WSNs for medical applications. A BSN may contain tens of sensor nodes with a specified sink node, distributed over a limited area such as an operation room or a nursing home. However, due to the hardware constraints and limited power supply, the range of wireless communication for each individual node is highly bounded. Thus a packet often has to be forwarded by a number of nodes to reach its destination. A concrete application scenario of BSNs is described in [11]. On an accident site difficult to access, there may be many injured persons and the available medics are limited. In such a situation a quickly deployed BSN on the accident victims may be used to collect and transmit vital sign data to a centralized medical server for diagnose and analysis so that proper and efficient medical operations can be carried out. For example, a sensor node may be used to measure the body temperature of an injured person with a certain period, and send the measured data to the sink node immediately or when it reaches a threshold value. Due to the life-critical nature of the application, certain QoS requirements on, e.g., network connectivity and packet delivery ratio must be guaranteed.

The difficulty in designing and analyzing a BSN is not in dealing with an individual sensor node in the network, which may be running a simple software. But as the network contains a number of nodes, and these nodes must cooperate to achieve some common goal, the behavior of such a network is extremely more complicated and difficult to analyze due to non-determinism. For instance, the network topology may be changing dynamically. The sensor nodes may move, disappear, and new nodes may appear from time to time. To achieve the common goal, the sensor nodes in a network need to follow a suitable communication protocol. The IEEE 802.15.4 [8] standard for wireless communication is one of such protocols. It offers different modes for communication and algorithms for packet routing if no direct connection to the sink exists. However, the specification of the standard covers only the logical behavior of a sensor node in wireless communication. Temporal configuration parameters, such as the active and standby period, of a node must be determined according to the application and the QoS requirements to be satisfied. For example, the application defines how often a sensor node should transmit measured data and the necessary bandwidth. The duration a node spends in the power down mode or in a mode for packet forwarding can also be carefully set to reduce energy consumption.



(a) (b)

Figure 1: Timing parameters and operation states of Chipcon CC2420 based sensor nodes.

Fig. 1(a) illustrates the main timing parameters associated with a sensor node. It has a main period covering three main modes: transmission, reception and power down (sleeping). The behavior of a node repeats over the main periods. It may, for instance, represent the measurement frequency of the sensor. The second main parameter is the active period. Within a main period, a node may stay active for some time and then switch to the power down mode. When it becomes active again, the node may begin to transmit data immediately or after a short delay. Within the active period, if a node is not transmitting data, it can receive data. The received data may need to be forwarded, which brings the node to the transmission mode again. The technical challenge here is to tune and validate the timing parameters such that the desired QoS requirements are satisfied. In a more complicated scenario, these parameters may be changing in an adaptive manner at runtime for each individual node. In this paper, we shall focus on the case of fixed parameters.

As an example, we study the Chipcon CC2420 transceiver [16] developed by Texas Instruments, which is widely used as the radio communication unit in sensor nodes. The chip implements wireless communication services for sensor nodes, following the IEEE 802.15.4 standard [8]. We shall develop a formal model using timed automata for the transceiver. A BSN based on such chips is modelled as a network of timed automata. The network studied allows dynamic reconfigurations of the network topology due to the physical movements of sensor nodes among fixed positions and also their temporally switching between active and inactive modes. We have used UPPAAL [10] to find the timing parameters and to validate QoS properties of the network. Both the simulator and model-checker of UPPAAL are used to analyze the average-case and worst-case behaviors. To demonstrate the usefulness of the technique, we have focused on packet delivery ratio and network connectivity. Our experiments show that even though the main feature of UPPAAL is model checking, it is also a promising and competitive tool for efficient simulation and parameter tuning. The simulator scales well; it can easily handle up to 50 nodes in our experiments. We have also shown how to formalize and check QoS requirements on network connectivity, end-to-end delay and packet delivery ratio using the UPPAAL query language. Compared with simulations, the model-checker may provide a guarantee on whether a requirement is satisfied by all possible behaviors of the network. Our experiments show that the model checker installed on a notebook with a Celeron 1.73 GHz processor and 1.5GB main memory is able to deal with BSNs of up to 16 nodes depending on the properties checked. These are BSNs of reasonable size for medical applications. Finally, to study the accuracy of our model and analysis results, we compare the simulation results by UPPAAL with traditional simulation techniques. The comparison shows that our analysis results coincide closely with simulation results by OMNeT++ [19], a widely used simulation tool for wireless sensor networks.

The paper is organized as follows. Section 2 provides a brief survey on existing validation techniques for WSNs. Section 3 describes briefly the behavior of transceivers in BSNs for wireless communication. In Section 4, we presents a timed automaton model for the Chipcon transceiver, and networks consisting of such transceivers. Section 5 shows how the model and UPPAAL are used for validation of QoS properties. Section 6 presents compares with traditional simulation techniques. Section 7 summarizes results and possible directions for future work.

2. RELATED WORK

Compared with classical simulation-based techniques, formal techniques are much less explored for the analysis of WSNs. Formal techniques have their limitation with scalability. But they can be used in the early design phase, e.g., to check the correctness of protocols and to identify worst-case scenarios for systems of moderate size. Automata-based techniques have also been used recently for analysis of wireless communication networks and protocols. In [6], a probabilistic timed automata model of the CSMA-CA contention resolution protocol according to the IEEE 802.15.4 standard is presented, and the PRISM tool is used to verify scenarios of data transmission in wireless networks. The work compares different configurations and abstractions of the model. In [5], the LMAC protocol is modelled in timed automata and a number of configurations for networks with four and five nodes are systematically analyzed using UPPAAL. However, to our best knowledge, there are no published works on validating the temporal parameters and QoS properties of WSNs using a model checker and comparing with existing simulation techniques for WSNs.

The WSN research community has developed numerous emulation tools such as Avrora [17], ATEMU [15] and COOJA [12]. An emulator provides a virtual operating environment to run the program (or with minor changes) written for a sensor node platform. For instance, Avrora can be used to emulate the execution of application program instruction-by-instruction at the level of clock cycle accuracy for AVR microcontroller based platforms, e.g., Mica2 sensor node. Detailed information about e.g., timers, radio, sensors and serial ports, and stack usage can be investigated, and the code can be tested and fine-tuned to achieve the best performance. Moreover, AEON [9], a tool built on the top of Avrora, can be used to evaluate the individual sensor node energy consumption and predict the lifetime of whole sensor networks. Emulators often focus on evaluating the behaviors of individual nodes. For the analysis of network level performance of WSNs, currently the most used validation techniques are based on Discrete Event Simulation. There exist well-developed simulators NS-2 and OMNeT++. These simulators have been further extended with accurate simulation models for various physical components and their access interfaces in WSNs, such as sensors and wireless channels, e.g., Castalia [14] based on OMNeT++ and SensorSim [13] based on NS-2. In these extended simulators, the simulation code (usually written in C or C++), defining the behavior of sensor nodes and wireless channel configurations, can be executed in the simulation environment. Due to the accurate modelling of physical components, these tools can be used to validate distributed algorithms and communication protocols in a realistic setting.

3. BIOMEDICAL SENSOR NETWORKS

The integration of biomedical sensors with wireless networks has led to the emergence of BSNs [7], which have great potential applications in medical care. In medical applications, body temperature, blood pressure, electrocardiogram (ECG), Pulse Oximeters (SpO2), and heart rate may be sensed and transmitted to a medical center, where the data is used for health status monitoring, and medical analysis and treatment. The main function of BSNs is to ensure that sensed medical data can be delivered to the medical center reliably and efficiently without physical wire-connections. Thus a BSN may contain a number of sensor nodes with a sink node collecting packets for the medical center.

3.1 The Chipcon CC2420 Transceiver

A sensor node usually consists of five parts: a microcontroller for data processing, sensor(s) for data collection, analog-to-digital converter (ADC) for signal conversion, a transceiver for wireless communication and a power supply unit. For the interoperability of sensor nodes from different manufacturers, IEEE Computer Society proposed the IEEE 802.15.4 standard [8] to define the protocol and compatible interconnection for data communication devices in WSNs.

One of the widely used hardware transceivers for wireless communication is the Chipcon CC2420 transceiver, developed by Texas Instruments according to the IEEE 802.15.4 standard. The CC2420 is a single chip designed for low-power and low-voltage wireless applications. It provides 250 kbps data rate with high receiving sensitivity (-95dBm). The reference manual of the CC2420 [16] defines the functionality of a CC2420 transceiver by a state machine. Fig. 1 (b) is an abstract version of the state machine with four abstract states. The state machine may be seen as the abstract behavior of a sensor node. The state transitions may be triggered by either command strobes or internal events, e.g., a timeout. Each of the abstract states represents a group of states in the original state machine. The Power-Down state combines the different energy saving states of a node, which may be entered from any state after the active period (see Fig. 1(a)) of the node has ended. The working states of a node during an active period are abstracted as RX for reception and TX for transmission. RX covers those states of a node, where it may be searching for a signal on the channel and can receive a packet at any time. The abstract state TX covers those states of a node for transmitter calibration, preamble, and frame transmission. If enabled, a node can acknowledge received packets. This functionality is represented by AckTX.

The transceivers in a network communicate with each other according to the protocols specified in the IEEE 802.15.4 standard, including routing, medium access control (MAC), and physical layer protocols. Routing protocols are used to define how data packets are delivered to the sink node through multi-hop communications. The physical layer is mainly responsible for data transmission and reception, the clear channel assessment (CCA) for carrier sense multiple access and collision avoidance (CSMA-CA), and activation (or deactivation) of the radio transceiver. The MAC sublayer handles all accesses to the physical layer channel and provides a reliable link between two peer MAC entities. For detailed information on these protocols, we refer to the IEEE 802.15.4 standard [8].

3.2 QoS Requirements

In medical applications where data packets usually contain vital medical information on human health, the network used for communication should guarantee that these packets are delivered to the medical center with a certain packet delivery ratio for a given time period. This is one of the most common QoS requirements on BSNs [4]. In this paper, we will focus on the following QoS requirements:

- *Network Connectivity:* Each node should have a connection with the sink node within a certain time period, either connected directly or through multi-hop communication. There should not exist isolated nodes.
- *Packet Delivery Ratio:* Packet loss can be caused by channel access failure, packet collision, transmission error caused by thermal noise and external interference. The packet delivery ratio for a given node is the ratio of the number of packets received successfully at the sink node by the number of packets sent by the node.
- *End-to-End Delay:* Data packets must be delivered to the sink node within a given time delay. The end-to-end delay is the time difference that a packet is ready to be sent at a sensor node until it reaches the sink node through multi-hop communications.

4. MODELLING BSNS WITH TIMED AUTOMATA

A timed automaton is a finite state automaton extended with real-time clocks. UPPAAL [10] is a tool box for timed automata, which provides a modelling language, a simulator and a model checker. In UPPAAL, timed automata are further extended with data variables of types such as integer and array etc., and networks of timed automata, which are sets of automata communicating with synchronous channels or shared variables, to ease the modelling tasks. The modelling language allows to define templates to model components that have the same control structure, but different parameters, which is a perfect feature for modelling of sensor nodes. For a tutorial of UPPAAL and timed automata, we refer to [3, 2].

In this section, we develop an UPPAAL model for a BSN, as a network of timed automata where each automaton models a sensor node. As all sensor nodes are implemented with the same chip for wireless communication, running the same protocol, we use a template to model the node behavior with open timing parameters to be fixed in the validation phase. The network topology is modelled using a matrix declared as an array of integers in UPPAAL. Elements in the matrix denotes the connectivity between pairs of nodes.

4.1 Modelling the Transceivers

Assume that the Chipcon CC2420 transceiver as described earlier is used for wireless communication in a sensor node. To study the network performance, we model the transceiver as an UPPAAL template based on the radio control state machine described in the reference manual [16].

The modelled template is shown in Fig. 2. For a detailed description of data, clock variables, names of states etc. used in the template, we refer to [18]. Most of the states are of the same name as the radio control states in the original state machine for the transceiver. The functionality of the transceiver is modelled by the state transitions according to the reference manual. The timing behaviors, as shown in Fig. 1, are formalized with clock constraints on transitions where the two important timing parameters, the main period (`P_M`) and the active period (`P_W`), are used as clock bounds.

In the real hardware, the main period will be started by an external signal from the sensor with a fixed period `P_M`. The signal indicates that there is a packet to send. We model this simply by a transition with a clock constraint enforcing the periodic behavior and a buffer assigned with the identity of the packet to be sent. Furthermore, in the real hardware, a node may send an acknowledgement after a successful reception of a packet depending on the configuration of the node. This is implemented implicitly by the dynamic routing scheme as described in the following subsection. Note also that we have added two extra states (i.e. `PreTX` and `Backoff`) to the part of the model concerning packet transmission. These states model the CSMA-CA back-off period in the communication protocol as described earlier.

4.2 Modelling the Network and Packet Transmission

The network topology – the spatial distribution of the sensor nodes – represents the direct connections between the nodes. It is the task of the routing protocol to find a path for a packet from one node to the sink. We model the network topology using a matrix (`topology`) referred as topology matrix. The dimensions of this matrix correspond to the number of nodes in the network. Every element stands for the connectivity from one node (row index) to another (column index). If the matrix should map the topology, negative values can be used, for instance, to represent that a pair of nodes is not connected and positive values can reflect the distance or signal strength between the corresponding nodes. The matrix can also be used to store routing information. In this case, some values can stand for a connection, where a node is in range but not on a routing path.

Using the topology matrix, it is easy to model a fixed routing scheme. The matrix also allows us to model dynamic reconfigurations of the network topology due to the movement of a node or the change of routing information at runtime. To study dynamic reconfigurations, we have modelled controlled flooding which is a dynamic routing scheme. A node broadcasts a packet to all its neighbors and remembers every received packet to control this flooding. If a node receives a packet that has been forwarded earlier, it will be ignored, which avoids cyclic forwarding. The model contains a matrix (`ignore`) with which every node remembers the packets it has received so far. The same matrix is used to remember if an acknowledgement is expected or received. In addition to dynamic routing, the flooding scheme offers the opportunity for an implicit acknowledgement: when a node has transmitted a packet, it will most likely receive it again after a short while, because the receiver(s) will broadcast it again. When a defined time after transmission has passed, a node will call a function (`ack`) to check if a packet has to be retransmitted.

To model packet transmission and transmission errors, we model only the transmission time given by the length of the packets, but abstract away from their contents. Every node has an unique identifier and if a node emits a packet, it is named by the identifier of the node. The identifier is also used to determine the length of the packet (`P_S[ID]`). To transmit a packet, a node uses a function named `send`. The function walks through the topology matrix and updates the incoming signal of every node in range, where the incoming signals are modelled by an array named `signal`. Packet collisions that lead to packet losses are modelled with help of the signal array. If a node starts a transmission while another node in range is receiving a signal, the corresponding element in the signal array will be set to a negative value meaning that the packet is corrupted.

5. VALIDATION USING UPPAAL

We consider a BSN as shown in Fig. 3, where the **S**-node is the sink node and the other nodes are modelled by the UPPAAL template presented above, with randomly chosen initial values for the timing parameters as listed in Table 1. The sink node is modelled as a simple automaton. It is not shown in the presentation as its essential behavior is only to accept packets from the other nodes and keep track of the number of packets received for each node. The network topology is chosen randomly. We shall study the network performance and show how to tune the timing parameters such that certain QoS requirements are satisfied.

5.1 Symbolic Simulation

Our goal is to use UPPAAL to simulate the behavior of

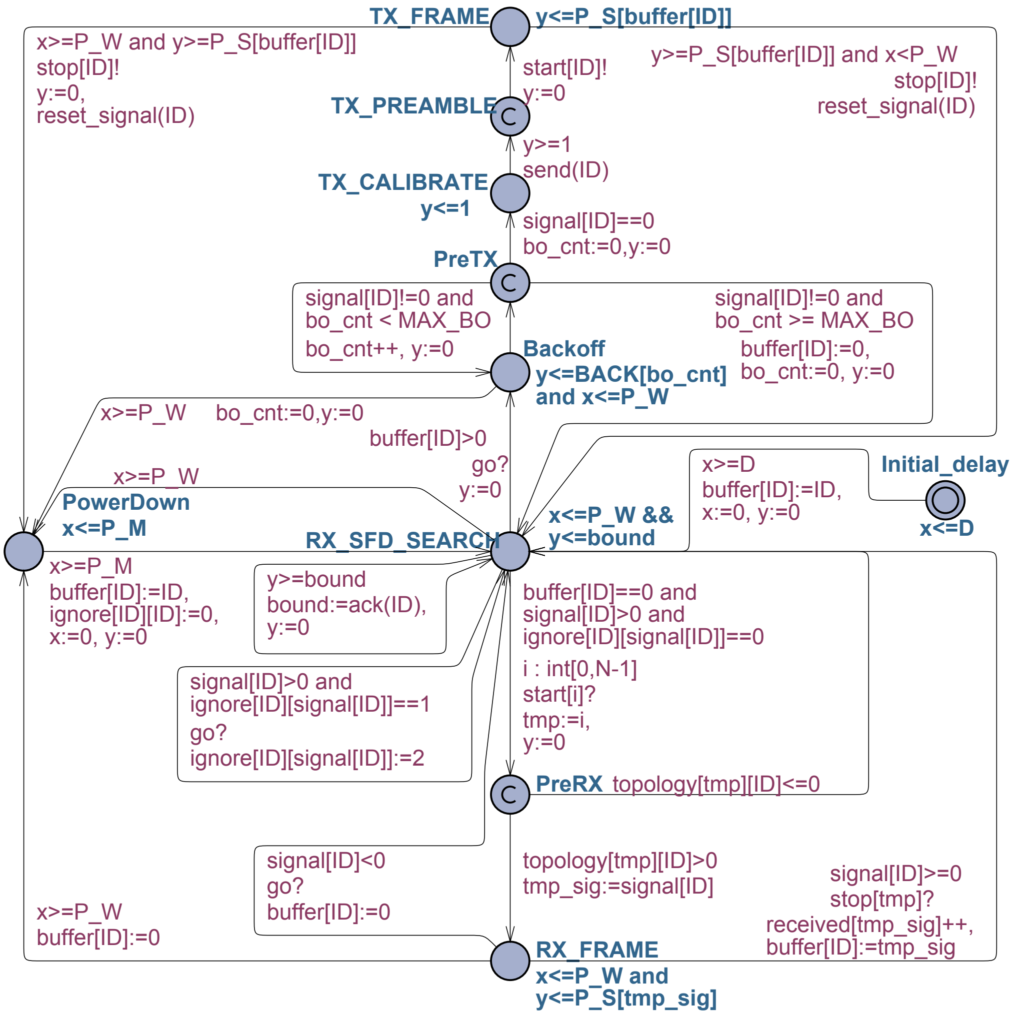


Figure 2: An UPPAAL template for wireless sensor nodes based on the Chipcon CC2420 Transceiver

the network based on the timed automaton model. For a detailed description of the UPPAAL simulator, we refer to [3, 2]. To enhance the scalability of the simulator, we have implemented a new version of the simulator optimized for exploring symbolic traces of models containing large data structures such as matrices. We have observed that the simulator scales well; for example we can easily handle networks with 50 nodes (and more), which is a big enough number for BSNs applications. However, for the presentation, we consider only the network shown in Fig. 3.

The simulator may be used to explore symbolic traces of a model. A symbolic trace is a sequence of transitions between symbolic states, corresponding to a collection of possible executions of the system modelled. A symbolic state contains the current control state, current values of data variables and possible clock values represented as clock constraints. As the symbolic states record all the changes of variables and clocks, they can be used to calculate performance metrics to validate all the QoS requirements summarized in section 3. For example to calculate the end-to-end delay for a packet, one may reset a clock in the original model to remember the time point when it is sent. When a symbolic state is found where the packet is delivered successfully, the bounds of the same clock in the found state represent the best- and worst-case delays for the packet.

Parameter Tuning

Now we show how to use the simulator to find timing parameters such that given QoS requirements are satisfied. We focus on simulations for packet delivery ratios. We simulate the network for 20.000 time units in which the nodes will complete between 65 and 110 main periods. The simulation takes about four minutes. The results are shown as diagrams in Fig. 4, where each curve illustrates the packet delivery ratio of a node, which is changing with time.

From the diagrams, we note that after a startup period, the packet delivery ratios for all nodes are stabilizing above a certain value, which indicates that the network performance is stable. For example, the packet delivery ratio stays above 80% for node 5, and 7 to 15, above 60% for node 6, above 50% for node 2 to 4, and under 40% for node 1, which is the worst of all.

Now if some or all the packet delivery ratios are not satisfactory according to the desired QoS requirements, we may tune the timing parameters of the nodes to influence or improve the network performance. Consider, for example, the

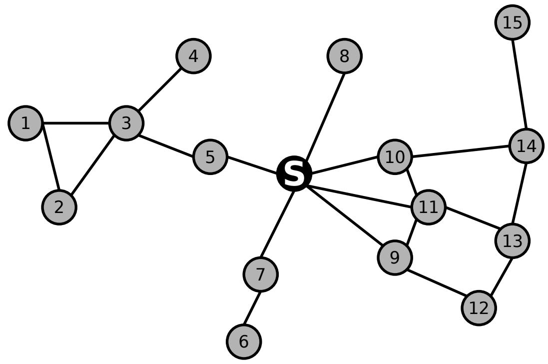


Figure 3: A random topology of a BSN with 15 sensor nodes.

Table 1: Timing parameters of the sensor nodes in Fig. 3.

	Initial Parameters		Improved Parameters	
Node	Main Period	Active Period	Main Period	Active Period
1	180	120	180	120
2	240	160	240	160
3	240	160	240	$\boxed{235}$
4	300	200	300	200
5	300	200	300	$\boxed{290}$
6	200	100	200	100
7	200	100	200	100
8	240	160	240	160
9	300	200	300	200
10	300	200	300	200
11	180	120	180	120
12	200	100	200	100
13	240	160	240	160
14	300	200	300	200
15	300	200	300	200

QoS requirement: "the delivery ratio for all nodes should be above 60%". If we compare the curves with the positions of the nodes on Fig. 3, we see that node 3 and 5 are bottlenecks for the connection of node 1,2, and 4 to the sink. So we may increase the duration of the active period of node 3 and 5, and hopefully these nodes will be able to forward packets most of the time.

The two new parameters for node 3 and 5 are given in Table 1 where the new ones are in boxes with the rest unchanged. With the new set of parameters, the packet delivery ratios from simulation are shown in Fig. 5. We notice an increase of about 10 up to 40 percentage points for the delivery ratio of node 1 to 4, and the delivery ratios for all nodes are stabilizing above 60% satisfying the above requirement. However, we should be aware that the increased active periods may lead to a higher energy consumption; one needs to find the right trade-off. In this paper, we will not consider QoS requirements on energy consumption.

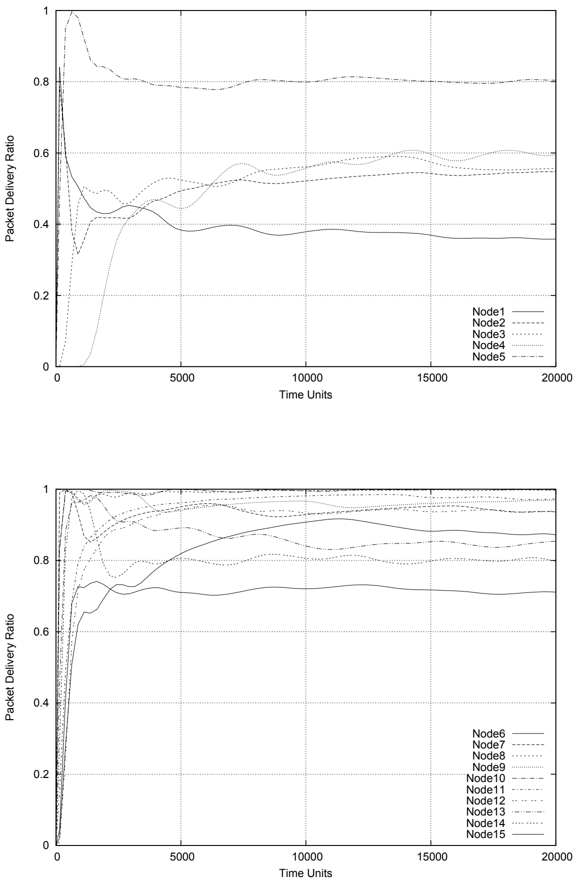


Figure 4: Simulation results for the network in Fig. 3 with initial timing parameters given in Table 1.

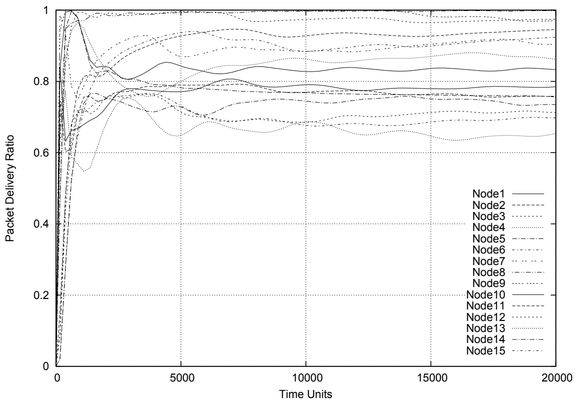


Figure 5: Simulation results for the network in Fig. 3 with improved parameters in Table 1.

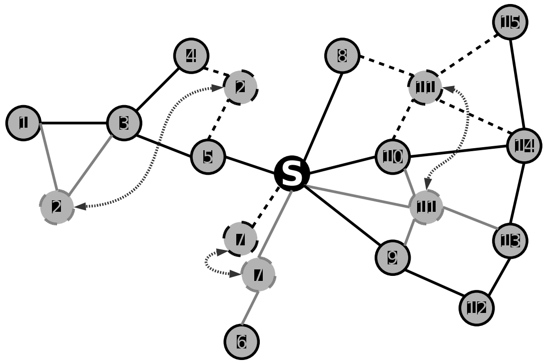


Figure 6: Example movements of mobile nodes.

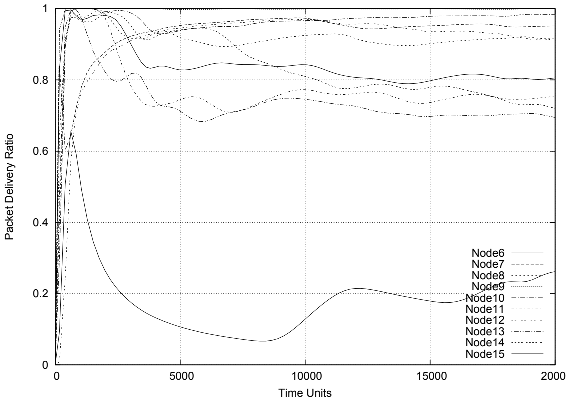


Figure 7: Simulation results for the network with mobile nodes as shown in Fig. 6 and initial parameters given in Table 1.

Dynamic Network Topologies

Note that the above simulations are dealing with dynamic network topology in the sense that, at any time point, some of the nodes may switch to the power-down state and disconnect some of the connections such that the network topology changes. The network topology may also change because of the movements of sensor nodes. To study the influence of this type of changes, we may use a timed automaton to manipulates the topology matrix to model the movement of mobile nodes.

We have studied the influence of movements of mobile nodes on the behavior of the whole network.

For example, we allow node 2, 7, and 11 to move according to Fig. 6. The simulation result is shown in Fig. 7. A comparison of Fig. 7 with Fig. 4 shows no essential differences in the packet delivery ratios for all nodes except for node 6. The reason is obvious: depending on the movement of node 7, node 6 may be isolated from all other nodes. We also see that the turning points follow the time points of movements for node 7.

5.2 Verification of QoS Properties

The networks we are dealing with are extremely nondeterministic; any node can communicate with any other node directly or indirectly at any time. With a simulator, we can explore only possible behaviors to study the average-case performance of a network. To reveal the worst-case scenarios and to check that some requirements are guaranteed by all possible behaviors of a network, we use the model checker of UPPAAL. However the goal here is not to show how powerful the tool is, rather to show that model checking is a useful technique to complement simulations. We shall use the UPPAAL query language [3, 2] to formalize QoS requirements concerning network connectivity and packet delivery ratio.

We have used UPPAAL installed on a notebook (with a Celeron 1.73 GHz processor and 1.5GB main memory) to check the formalized requirements. The model checker can handle networks with 5 up to 16 nodes depending on the properties to be checked. The verification results are summarized in Table 2. More examples of QoS requirements verified can be found in [18]. We note that for most of the requirements listed, the verification times are within minutes.

Network Connectivity

As described in Section 3, for BSNs, we are interested in network connectivity to guarantee that each node is connected with the sink node. For this purpose, in the model, we have used an array `received`. Each element of the array (initialized with 0) is a counter associated with a node and incremented whenever the sink receives a packet emitted by the according node. For a node with identity X, we use the query `A<>received[X]>0` to prove or disprove, whether X can eventually establish a connection to the sink. This allows us to find improper timing parameters which result in that some nodes are isolated.

Note that `A<>received[X]>0` states that there will be a connection eventually without a time bound. To estimate the maximal delay, we use the number of main periods of node X. We modify the model such that the counter `periods[X]` is reset whenever the sink receives a message from node X. Then we can use the query `A[]periods[X]<Y` to prove that node X is connected to the sink at least within Y periods. The array for the number of received packets has no impact on the properties verified here and thus it can be declared as a meta variable, i.e. that these variables do not extend the number of states that have to be explored during verification.

Table 2: Example verification results.

Property	Network Size	CPU Time (Sec)	Memory (MB)
Connectivity	11 Nodes	315.67	73.45
Bounded Connectivity	6 Nodes	157.71	36.7
Packet Delivery Ratio	6 Nodes	252.80	38.6

Packet Delivery Ratio

Recall that the packet delivery ratio of a node is the ratio of the number of packets delivered to the sink by the number of packets sent from the node, and the later is the number of main periods. These numbers are denoted by the counters `received[X]` and `periods[X]` in the model. Ideally we may want to check that over time, the packet delivery ratio of certain packets is over 90%. Unfortunately, in UPPAAL we can not use the query language to specify such properties concerning mean values or duration properties. However, we may run a number of checks to approximate the packet delivery ratio. We may check at least N out of M packets sent will be delivered successfully using the query, `A[]periods[X]>=M imply received[X]>=N`. For instance, for ten packets sent, we may check whether a packet delivery ratio of at least 90% is reached using the query `A[]periods[X]>=10 imply received[X]>=9`. We reset `periods[X]` and `received[X]` when the bounds are reached to assure that the property is not only satisfied after the first ten periods, but whenever ten periods have been completed. We may change the bounds on the numbers of packets sent and received to achieve better approximations.

End-to-End Delay

For each packet, we may associate a clock which is reset when the packet is sent and then check the lower and upper bounds of the clock when the packet is delivered. We may get a lower bound in this way, but as the packet may be lost the upper bound will be infinity in general.

However, we can indeed induce an upper bound from the analysis result on packet delivery ratio. For example, if one out of two packets sent will be delivered successfully, the worst case delay is bounded by the length of two main periods. Note that it is assumed that every main period, a sensor node will send one packet. Thus if some important data is twice in two packets, the data will be delivered for sure within two main periods.

6. COMPARISON WITH DISCRETE EVENT SIMULATION

One of the main concerns in applying model-based techniques is to develop faithful models of systems to obtain faithful analysis results. To study the accuracy of our model, we compare simulation results by UPPAAL with the traditional discrete event simulator OMNeT++, which is widely accepted in the WSN community. We shall see that for two typical application scenarios of BSNs, the UPPAAL simulation results for packet delivery ratio using our model coincide closely with simulation results by OMNeT++. However, we have also observed some minor differences due to the simplifications in the modelling of packet transmissions and collisions.

6.1 Simulation Settings

To ease the comparison, we study two fixed network topologies as shown in Fig. 6.2(a) and 6.2(b), corresponding to two typical application scenarios of BSNs in medical care. The first topology for networks with one-hop communication is usually used for in-field patient monitoring where the sensor nodes are deployed in a small area, while the second for multi-hop communication is often used in a large area, where data packets cannot be transmitted to the medical server directly.

In the study with OMNeT++, we use the Castalia WSN simulator [14]. The WSN simulator is configured to follow the physical layer and MAC sublayer protocols as defined in the IEEE 802.15.4 standard. For sensor nodes, we adopt two types of sensors, ECG and temperature sensors with fixed sampling rate and packet size, which are often used in medical care. The ECG sensors emit 5 packets with a size of 100 bytes every second, and the temperature sensors emit 1 packet with a size of 2 bytes every second.

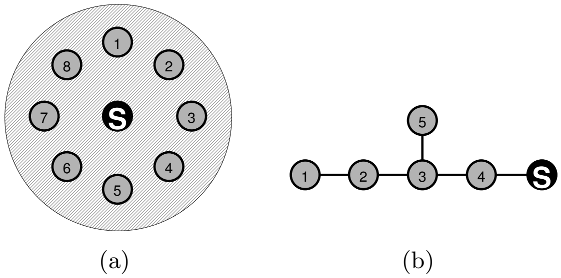
In the study with UPPAAL, the timing parameters including the main periods and transmission delays of our model for the Chipcon CC2420 transceiver are initialized according to the sampling rate and packet size of ECG and temperature sensors. The topology matrix is fixed according to the network topologies shown in Fig. 6.2(a) and 6.2(b).

Table 3 lists all necessary parameters for simulating sensor networks that are IEEE 802.15.4 compliant. These parameters are used in the configuration of the WSN simulator for OMNeT++, and our transceiver model for UPPAAL simulation. In particular, macMinBE is the initial value of backoff exponent, aMaxBE is the maximum number of backoff exponent, and macMaxCSMABackoffs is the maximum number of backoffs that the CSMA-CA algorithm will attempt before declaring a channel access failure. Note that for simplicity, we have used a fixed *packet overhead* for all packets on physical, MAC and application layer. Note also that for UPPAAL simulation, we have used a simplified collision model – listed as *simplified* in the table – meaning that if more than one sensor nodes within communication range transmit simultaneously, collision happens and all the packets are corrupted.

Table 3: Simulation parameters

Parameters	**OMNeT++**	**UPPAAL**
channel model	log shadowing wireless	fixed bit error rate
path loss exponent	2.4	N/A
collision model	additive interference	simplified
data transmission rate	250 kbps	250 kbps
simulation time	300 s	300 s
packet overhead	32 bytes	32 bytes
macMinBE	3	3
aMaxBE	5	5
macMaxCSMABackoffs	3	3

6.2 Experiment 1: one-hop Communication



(a) (b)

Figure 8: The different topologies that were used to compare simulation results obtained by OMNeT++ and UPPAAL.

We fix a one-hop communication network with the topology shown in Fig. 6.2(a). In the network, all nodes including the sink node are one-hop neighbors to each other, which means that each node is connected directly to the sink node and all nodes can communicate with each other with one-hop communication. Node 1, 3, 5, 7 are ECG sensors and node 2, 4, 6, 8 are temperature sensors. Node S is the sink node to collect data packets sent by the sensors at fixed rate as described above.

Table 4 lists the Packet Delivery Ratio for each node according to OMNeT++ and UPPAAL simulations, respectively.

Table 4: Packet Delivery Ratios from Simulations with OMNeT++ and UPPAAL for the one-hop communication network shown in Fig. 6.2(a).

Sensor	OMNeT++ (in %)	UPPAAL (in %)
Node 1	96.1	99.6
Node 2	96	95.7
Node 3	96.2	99.7
Node 4	95	99.0
Node 5	95.7	99.6
Node 6	96.3	97.3
Node 7	95.9	99.7
Node 8	96.7	97.7

6.3 Experiment 2: Multi-hop Communication

Fig. 6.2(b) shows a multi-hop communication network. Node S is the sink node, node 1 and 2 are ECG sensors, node 3, 4 and 5 are temperature sensors, respectively. Data packets are routed to the sink from node to node by multi-hop transmission.

Table 5 lists the packet delivery ratio for each node according to simulations by OMNeT++ and UPPAAL respectively.

Table 5: Packet Delivery Ratios from Simulation with OMNeT++ and UPPAAL for the multi-hop communication network shown in Fig. 6.2(b).

Sensor	OMNeT++ (in %)	UPPAAL (in %)
Node 1	75.2	97.5
Node 2	84.4	97.7
Node 3	91	97.7
Node 4	95	100
Node 5	86	92.8

6.4 Comparison and Discussion

In the first experiment, the results of delivery ratio for each node from UPPAAL and OMNeT++ are similar, indicating that the UPPAAL model is faithful in capturing the behavior of the CC2420 transceiver compared with OMNeT++ simulations. In the second experiment, for both OMNeT++ and UPPAAL, the delivery ratios for node 4 (one hop to sink) are 95% and 100% respectively. Other nodes through multi-hop communication perform worse than node 4 with only one hop. The main difference is that in OMNeT++ the performance of node 1 is the worst, while in UPPAAL the performance of node 5 is the worst. Again the main reason is that the simplified model of wireless channels in UPPAAL, makes the delivery ratio of a node depend less on the number of hops to the sink; whereas the wireless channel is modelled in a more realistic way than that in OMNeT++. This is explained as follows.

In a WSN, whether a packet can be received successfully by a receiver depends on the Signal to Noise Ratio (SNR), which depends dynamically on the transmitting power, receiving sensitivity, thermal noise, propagation loss, and interferences. For multiple concurrent transmissions, even nodes that are not in the same communication ranges may interfere each other by increasing the background noise level. That is, there are still interferences even though no collision occurs.

For OMNeT++, the results are quite reasonable. Node 1 needs four hops to the sink, during the relaying procedure, packet loss can be caused by channel access failure, collision, and interference from multiple concurrent transmissions. Multi-hop has significant impact on the performance of packet delivery ratio. This explains why the delivery ratio for node 1 with four hops to the sink is not as good as node 5 with one hop less.

For UPPAAL, we observe that the delivery ratio for node 1 is better than for node 5. The reason is as follows: as the model for wireless channels considers only collision and fixed transmission bit error, propagation loss and interferences due to concurrent transmissions are abstracted away, the channel appears to be perfect if there is no collision. Thus, multi-hop communication has less impact on network performance than it should be. However, the possibilities of collision are modelled close to reality and thus have a relatively higher impact on the packet delivery ratio. From the network topology, we see that collision occurs most likely because node 5 sends a packet to node 3 simultaneously with node 2, which is either transmitting its own packet or forwarding a packet of node 1. Because this collision can only happen, when node 5 is transmitting a packet, node 5 will lose the same number of packets as node 1 and 2 together. Due to the different frequencies for packet emissions of node 1, 2, and 5, within the same time period, node 1 and 2 emit considerably more packets than node 5, thus the number of lost packets of node 5 has a higher impact on the according packet delivery ratio than the number of lost packets for node 1 and 2.

From the simulation results, we may conclude that our model for the CC2420 transceiver is reasonably accurate compared with the WSN simulator of OMNeT++. While the wireless channel model should be improved in UPPAAL – especially for networks with multi-hop communication. However, since the simulation requirements and applied situations are different, certain abstraction must be made to achieve the trade-off between accuracy and verification capabilities.

7. CONCLUSIONS AND FUTURE WORK

The main contributions of this paper include: (1) We have developed a formal model using timed automata for the Chipcon CC2420 transceiver, which is one of the most used hardware chips for wireless communication in sensor networks. To our best knowledge, this is the first model for such transceivers. We believe that the model can be extended easily to model and validate other transceivers (or wireless communication devices), and communication protocols, that are not necessarily limited to be IEEE 802.15.4 compliant. (2) We have shown how to use the UPPAAL tools to tune and validate the timing parameters of the sensor nodes such that the desired QoS requirements are satisfied. (3) To study the accuracy of our model and analysis results, we have compared the simulation results by UPPAAL with traditional simulation techniques using OMNeT++, a widely used simulation tool for wireless sensor networks. The comparison shows that our analysis results coincide with simulation results by OMNeT++, that are different only due to the simplified channel model in UPPAAL. We also observed that using OMNeT++, it is very time-consuming to implement the simulation code in C++, which has the advantage to be more precise for simulating low-level implementation details; whereas with UPPAAL simulations, one can easily tune the model to study network-level performance in the early design phase.

As future work, we shall study the other types of QoS requirements on energy-consumption and bandwidth as well as jitter i.e., the variation of delay experienced by the sink node. We shall also investigate the mobility degree of a BSN and its impact on QoS properties. An extension of the channel model used in the UPPAAL systen is an important issue to improve its accuracy, especially for properties like packet delivery ratio. Another interesting direction for future work is to develop a logic and extend the UPPAAL model checker to fully capture QoS requirements studied in this paper and the other requirements on energy consumption and network throughput for medical applications [4]. The challenge is to deal with properties concerning mean values such as "over the life time of a network, the energy-consumption per time unit is within a given bound".

8. REFERENCES

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

[2] G. Behrmann, A. David, and K. G. Larsen. A tutorial on UPPAAL. In M. Bernardo and F. Corradini, editors, *Formal Methods for the Design of Real-Time Systems: 4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM-RT 2004*, number 3185 in LNCS, pages 200–236. Springer–Verlag, September 2004.

[3] J. Bengtsson and W. Yi. Timed Automata: Semantics, Algorithms and Tools. *Lecture Notes on Concurrency and Petri Nets*, LNCS 3098:87–124, 2004.

[4] D. Chen and P. K. Varshney. QoS support in wireless sensor networks: A survey. In *Proc. of the 2004 International Conference on Wireless Networks (ICWN'04)*, pages 227–233, Las Vegas, Nevada, USA, June 2004. CSREA Press.

[5] A. Fehnker, L. F. W. van Hoesel, and A. H. Mader. Modelling and verification of the lmac protocol for wireless sensor networks. Technical Report TR-CTIT-07-09, Centre for Telematics and Information Technology, University of Twente, Enschede, February 2007.

[6] M. Fruth. Probabilistic model checking of contention resolution in the ieee 802.15.4 low-rate wireless personal area network protocol. In T. Margaria, A. Philippou, and B. Steffen, editors, *Proceedings of the 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2006)*, Paphos, Cyprus, November 2006.

[7] Y. Guang-Zhong, editor. *Body Sensor Networks.* Springer, New York, 2006.

[8] IEEE Standard 802.15.4. *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, 2003.

[9] O. Landsiedel, K. Wehrle, B. Titzer, and J. Palsberg. Enabling detailed modeling and analysis of sensor networks. *Praxis der Informationsverarbeitung und Kommunikation*, 28(2):101–106, April 2005.

[10] K. G. Larsen, P. Pettersson, and W. Yi. UPPAAL in a Nutshell. *Int. Journal on Software Tools for Technology Transfer*, 1(1–2):134–152, October 1997.

[11] X. Liang, B. Østvold, W. Leister, and I. Balasingham. Credo: Modeling and analysis of evolutionary structures for distributed services – user driven requirements, March 2007. Diliverable D6.1, EU IST project, number 33826.

[12] F. Osterlind, A. Dunkels, J. Eriksson, N. Finneand, and T. Voigt. Cross-level sensor network simulation with COOJA. In *Proc. of the 31st IEEE Conference on Local Computer Networks*, pages 641–648, Tampa, Florida, USA, November 2006.

[13] S. Park, A. Savvides, and M. B. Srivastava. SensorSim: a simulation framework for sensor networks. In *Proc. of the 3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems (ACM MSWiM 2000)*, pages 104–111, Boston, Massachusetts, USA, August 2000.

[14] H. N. Pham, D. Pediaditakis, and A. Boulis. From simulation to real deployments in WSN and back. In *Proc. of the 8th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'07)*, pages 1–6, Helsinki, Finland, June 2007.

[15] J. Polley, D. Blazakis, J. McGee, D.Rusk, and J. Baras. ATEMU: a fine-grained sensor network simulator. In *Proc. of the 1st IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON'04)*, pages 145–152, Los Angeles, California, USA, October 2004. IEEE Press.

[16] Texas Instruments Inc. *2.4 GHz IEEE 802.15.4 / ZigBee-Ready RF Transceiver (Rev. B), CC2420 data sheet*, March 2007.

[17] B. L. Titzer, D. K. Lee, and J. Palsberg. Avrora: scalable sensor network simulation with precise timing. In *Proc. of the 4th International Symposium on Information Processing in Sensor Networks (IPSN'05)*, pages 477–482, Los Angeles, California, USA, April 2005. IEEE Press.

[18] S. Tschirner, L. Xuedong, and W. Yi. Model-based validation of qos properties of biomedical sensor networks. Technical report, Department of Information Technology, University of Uppsala, 2008.

[19] A. Varga. The OMNeT++ discrete event simulation system. In *Proc. of the 15th European Simulation Multiconference (ESM'01)*, pages 319–324, Prague, Czech Republic, June 2001. SCS.