The tool for modeling of wireless sensor networks with nested Petri nets

Nina Buchina
Department of Software Engineering
National Research University Higher School of Economics
Moscow, Russia
m.e.tigra@gmail.com

Leonid Dworzanski
Department of Software Engineering
National Research University Higher School of Economics
Moscow, Russia
leo@mathtech.ru

The work describes a specific approach to modeling and simulating Wireless Sensor Networks (WSN), using nested Petri Nets formalism. The tool for modeling/simulating WSN must take into consideration resources, time and sensors cost. Even though Petri Nets are good for modeling dynamic systems, they do not have enough means to express these aspects. Hence some extension were introduced to handle them.

Keywords—wireless sensor networks; petri nets; modeling; simulating; analysis;

I. Introduction

Two main concepts are discussed in this paper that reader may need to be acquainted with in advance. These are Wireless Sensor Networks and Petri Nets.

A. *Wireless sensor networks*

Wireless sensor networks are computer networks that consist of distributed sensor nodes. These sensors are used to monitor physical or environmental conditions, such as temperature, sound, pressure and others, and pass the information to the main host (server, end point). These networks are subject of active study, and are utilized in such systems like smart houses, security, environmental and others.

For example, an environmental wireless network described in [3] detects different wild animals as they go through special passages under roads. It also addresses to identification problem. This task is fulfilled by filling the space within some radius from wildlife passages with the wireless nodes of different types and some wireless cameras. In this way scientists may not only find out that the animals have passed the camera, but they also will know the path and direction they followed.

Another widespread usage of WSN are road traffic control and optimization systems. The paper [4] presents a project aimed to "provide a wireless solution for obtaining traffic-related data that can be used for automatically generating safety warnings at black spots along the road network". The WSN consists of numerous sensors deployed along the road. Each of them monitors its road section, collects parameters such as number, speed and direction of vehicles.

Nets of these types are difficult and expensive to develop and to build. Moreover, the price of possible mistake and error can be extremely high. It may also turn out that the network in principle cannot operate as expected. And as for any complex system - the sooner this fact will be discovered, the higher are chances to fix the situation before it will drain all the budgets.

This is why modeling and simulating are extremely required for large-scale and expensive networks. There is also a possibility that the deeper understanding of the network will be gained, and the better network structure will be discovered in the process of network modeling.

B. *Petri nets*

Petri nets is a classical formalism also known as a place/transition net. Petri Nets are commonly used to model different sorts of structures, processes and systems. It also has many extensions as the concept has been adapted for different needs. In this work, two of such extensions will be used. These are Nested Petri nets [1] and Time Petri Nets [2].

It should be outlined that in this work we assume that all the messages in Wireless Sensor Network are intended to end up on the server sooner or later. There may exist specific network that enables sensor nodes to exchange meaningful messages between each other, but by now they are not addresed here.

II. Related work

A. *Wireless sensor network modeling and simulating*

There are currently plenty of works concern modeling sensor networks or simulating them.

First of all, there are several mature software products for modeling and simulating networks. Almost all of them share the same drawback: they are very general as they allow to simulate almost any kind of network, and thus are hard to learn and to use for specific needs:

– OPNET [6]

– NetSim [7]

– ns-2 [8]

– OMNet++ [9]

There are also many works performed by researchers, aiming to survey existing network modeling approaches or create new modeling tools.

One of these tools is Shawn[10]. Its authors state that the tool is designed to support several implementation models and propose an algorithm for developing distributed implementation

The research is partially supported by the Russian Fund for Basic Research (project 11-01-00737-a).

model of the network. One of main goals of Shawn is to provide support for extremely large networks.

Another wireless sensor network simulating tool that emerged lately is NetTopo [11]. This tool deals with not only WSN model, but also with a real-world wireless sensors testbed. The simulation of virtual WSN, the visualization of real testbed, and the interaction between simulated WSN and testbed are its key challenges.

The review paper [12] summarizes the state of the problem at 2005, which stays mostly relevant.

B. *Petri nets and wireless sensor networking*

Petri nets are usually taken by some other scientists as a base to create their own modeling language and verification framework.

One of recent works [13] proposes the concept of intelligent wireless sensor networks model (IWSNM) based on Petri nets, which can accurately and unambiguously model the overall and individual characteristics of the networks. Moreover, IWSNM can be analyzed, verified and validated by the supporting tools of Petri nets.

Another work [14] by computer scientists from the University of Virginia proposes an approach to formal application-level requirements specification in wireless sensor networks. They present an event specification language that supports key features of WSNs. As a description language, it is an extension of Petri nets. It integrates features ranging from color, time, and stochastic Petri nets to tackle problems in specification and analysis.

III. GOALS

As it has been shown in the previous section, there are plenty of tools and works intended to help dealing with wireless sensor networks by modeling and simulating them. While commercial products like OPNET can be quite expensive, the academic tools require considerable amount of time and efforts to learn.

On the other hand, a wireless sensor network modeler that is completely free to use and extremely easy to operate would empower more enthusiasts to design their own sensor networks. They would have an opportunity to ensure their projects are actually viable before trying to implement them. Involvement of amateurs may help to accelerate the development of wireless sensor networks industry.

Therefore, the main practical goal of this work is to build a wireless sensor network modeling tool, consisting of 3 modules:

1. Graphical User Interface;
2. The modelling kernel;
3. Analysis backend.

The tool should support some analyzing options such as

1. Liveness
2. Deadlock-freedom
3. Safety

All these are to be done via converting the WSN graph to a Petri net and analyzing this net with standard techniques.

The tool must also be user-friendly and extremely easy to use. It should be possible to perform basic operations even if the user has no deep knowledge on networking or Petri nets.

The theoretical goal of the work is to widen nested Petri notation with WSN-specific features, such as:

1. Time
2. Resources
3. Cost

The last goal is to use the created tool to model and simulate a real wireless sensor network, estimate its characteristics, such as cost, performance etc., and decide whether it can be implemented (and whether it is reasonable).

IV. RESULTS ACHIEVED

A. *Going Petri*

A conversion scheme was created to translate a set of wireless sensor nodes and their coordinates to the nested Petri net. This scheme consists of 2 parts:

1) Generation of overall net model that shows which nodes can communicate to which, and what ways there exist to pass the signal to the server. In general, each sensor node corresponds to two Petri places connected with transition. One of these places is called "implementation place" and is used to store tokens that represent the wireless node implementation. The other place is called a "signal place". If 2 wireless nodes in real world are able to communicate, their corresponding signal places are connected with Petri transitions like so:

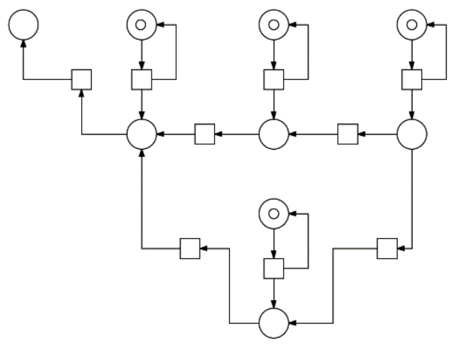

Fig. 1 Sample Petri Net generated from wireless sensor net model. Here each implementation place contains a token.

2) Each WS Node has its own Petri model that describes its behavior. It's network developer's task to describe the behavior of his wireless nodes. However, the tool will have a bunch of "standard" nodes with predefined behavior. The node's Petri model is stored in a token that is placed to node's corresponding implementation Petri place.

The signal is represented by a Petri token. This token travels through the signal states of the net to the server.

Communication is done via horizontal and vertical synchronization [1]. For example, the transition between Node place to its signal place is enabled only when some state of node implementation (stored in token in the implementation place) is achieved.

B. *Routing*

The order in which nodes participate (or do not) in message transmitting depends on the selected routing algorithm. Currently the tool supports only "least hops" algorithm [5].

The routing is performed on the stage of conversion between WSN structure and Petri net. If two nodes can be used for message transmission, the tool will add a Petri transition to the state that corresponds to the node with least hops number. If there are two nodes with least hops number, the tool will add create both transitions.

This algorithm is about to change when time concept will be introduced. Then no transmission possibilities will be ignored; time windows will be applied for each generated transition instead (see Future work directions, item E: Time problems)

C. *The Tool*

The tool at its current state allows its user to arrange wireless nodes using simple GUI, and then create a Petri net out of this model with respect to routing.

The tool is written fully on Java and consists of 4 main components.

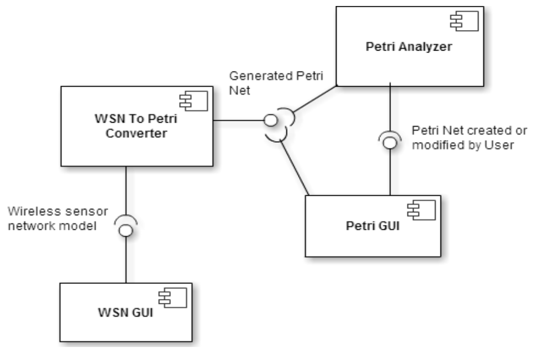

Fig. 4 The component diagram of the tool

The first component is WSN to Petri converter. It's only purpose is to convert the wireless sensor network model defined by user to its Petri representation with respect to routing algorithm.

The main module of the tool is the Analyzer. It is designed to contain a set of verification algorithms that can be applied to Petri net or Petri-based model. The exact list of algorithms is still being composed, but it is sure to contain tools for analyzing reachability, deadlocks and time characteristics of the net.

The last module is graphical user interface. It actually consists of two main parts.

The first part lets user to arrange wireless sensor nodes on a map or floor plan, which is being uploaded to the tool by the user. The wireless sensor network model is created as a result.

The second part is designed to represent Petri net generated by the tool or created by user. The presented net can also be modified by the user, except for the case of the Petri Net that was generated from wireless sensor network model. In this case it seems more reasonable to modify the original wireless sensor model and to generate its Petri representation again.

On screen-shots below a sample small wireless sensor network is shown, and so is its Petri representation.

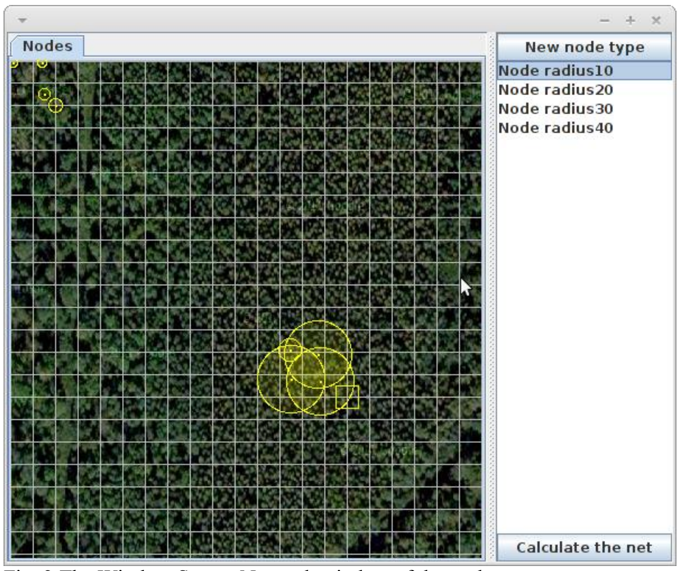

Fig. 2 The Wireless Sensor Network window of the tool.

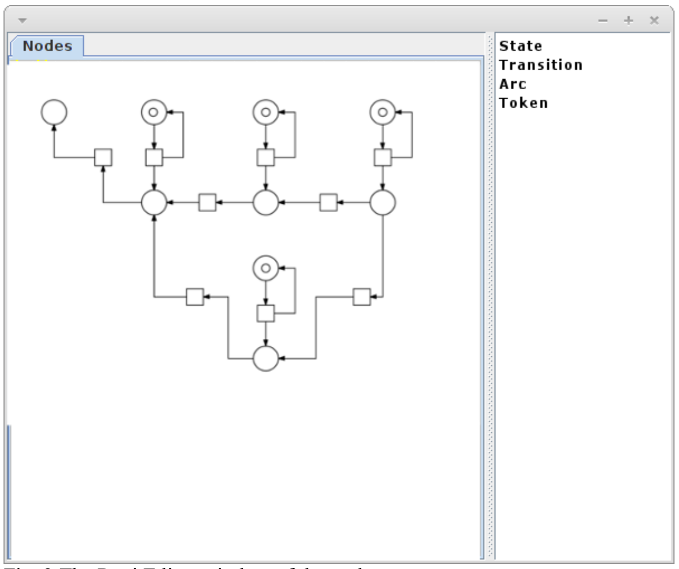

Fig. 3 The Petri Editor window of the tool.

V. Future Work Directions

C. *Fully implement the tool*

This point assumes implementing all the points from the "Goals" section. It also will require testing the tool against some real WSN and comparing tool's outputs and prognosis with the real situation.

D. *Add support of data packages*

At this time, each message is represented by a single token. It would be reasonable to allow user to simulate different message sizes to estimate network bandwidth. For example, each Petri token can be said to represent 1 KB of data. This work will also require to control each message integrity.

E. *Develop more predefined nodes*

This point will require creating Petri models of some popular wireless sensor nodes, and introduce these in the tool as representation of wireless sensors behavior. The user of the tool will then be able to edit these models and adapt them for her needs.

F. *Analysis options*

Some examples of analysis that theoretically can be done using Petri representation of WSN is given in the Goals section. The task is to recognize as many such options as possible and present the results in form of article, also implementing in a tool.

G. *Enable the support of decentralized networks*

Implementing this feature will allow to model and simulate networks with multiple message end-points (servers), with no servers at all, or with nodes able to consume messages, not just transmit them.

H. *Time problems*

Introducing the Time to model lets researcher to deeply explore performance of his network. It also lets him to address the problem of network liveness, as the net may reconfigure itself when some nodes become unavailable after certain timeout.

In order to address time problems it has been decided to use Time Petri Net formalism. This concept is described in detail in [2].

This work introduces the concept of time windows. The transition may be fired only if the time elapsed since its last enabling gets into certain interval – the time window. This concept may be used to implement WSN reconfiguration: once the node that is supposed to transmit the message is identified as not responding (the time went out) the net should try to pass the message using a different route.

I. *Add multiple routing algorithms support*

There are many more possible routing options than just counting hops. The user may prefer routing based on availability, performance, delay time and other characteristics.

The task so far is to add some popular routing algorithms support in a tool. Anyway, it also seems to be reasonable to allow users to create their own routing algorithms. This may be achieved by providing them a java interface to implement and enabling the tool to use these implementations along with build-in routing algorithms.

REFERENCES

[1] И. А.Ломазова "Вложенные сети Петри: моделирование и анализ распределенных систем с объектной структурой". Научный мир 2003, Москва

[2] L. Popova-Z. "Time and Petri Nets - A Way for Modeling and Verification of Time-dependent Concurrent Systems". Humboldt-Universität zu Berlin Department of Computer Science December 2012, Moscow

[3] A.-J. Garcia-Sanchez, F. Garcia-Sanchez, F. Losilla, P. Kulakowski, J. Garcia-Haro, A. Rodríguez, J.-V.López-Bao, F.Palomares. "Wireless Sensor Network Deployment for Monitoring Wildlife Passages". Sensors 2010, 10, 7236-7262;

[4] M. Franceschinis, L. Gioanola, M. Messere, R. Tomasi, M. A. Spirito, P. Civera. "Wireless Sensor Networks for Intelligent Transportation Systems". http://www-mobile.ecs.soton.ac.uk/home/conference/vtc09spring/DATA/07-02-04.PDF

[5] Bellman, Richard (1958). "On a routing problem". Quarterly of Applied Mathematics 16: 87–90. MR 0102435

[6] OPNET modeler website: http://www.opnet.com/solutions/network_rd/modeler.html

[7] NetSim website: http://tetcos.com/

[8] ns-2 wiki: http://nsnam.isi.edu/nsnam/index.php/Main_Page

[9] OMNet++ website: http://www.omnetpp.org/

[10] Shawn – a customizable sensor network simulator. https://www.itm.uni-luebeck.de/ShawnWiki/index.php

[11] Lei Shua, Manfred Hauswirthb, Han-Chieh Chaoc, Min Chend, Yan Zhange: "NetTopo: A framework of simulation and visualization for wireless sensor networks".Ad Hoc Networks 9 (2011) 799–820

[12] E. Egea-López, J.Vales-Alonso, A. S. Martínez-Sala, P. Pavón-Mariño, J. García-Haro: "Simulation Tools for Wireless Sensor Networks". Summer Simulation Multiconference - SPECTS 2005

[13] X Fu, Z Ma, Z Yu, G Fu: "On Wireless Sensor Networks Formal Modeling Based on Petri Nets"

[14] Binjia Jiao Sang H. Son John A. Stankovic: "GEM: Generic Event Service Middleware for Wireless Sensor Networks"