Umeå University
Department of Computing Science

Object-Oriented Programming Methodology 7.5 p
5DV133

OU4 Sensor Network

Submitted
2016-05-23

Authors:
Johan Eklund (kv03jed@cs.umu.se)
Tommie Lindberg (c15tlg@cs.umu.se)
Jakob Lundin (c14jln@cs.umu.se)
Lorenz Gerber (dv15lgr@cs.umu.se, lozger03@student.umu.se)

Instructors:
Anders Broberg
Niklas Fries
Adam Dahlgren
Jonathan Westin
Erik Moström
Alexander Sutherland

Contents

1 Introduction

The assignment was described on the course homepage [2]. The aim was to implement software that allows to perform experiments on sensor networks as described in Braginsky and Estrin [1]. The main topic of [1] is the use of *rumour routing* as an energy saving message transportation algorithm that for example be used in environment surveillance networks.

The physical setup to be modelled consists of a number of autonomous sensor nodes with a limited signal send/receive range. Simplified it can be said that a networking algorithm shall defome how to spread information of a detected event from nodes of event origin to the rest of the network. Two simple algorithms to either spread or demand information is spoofing. Here for example after detecting an event, the node sends the information further to every to him known node, and the next node will to the same. This will result in a large amount of redundancy hence it is not very energy saving. The same is true for requesting information about an event in this way, by sending out an information request to every known node.

Braginsky and Estrin [1] propose an algorithm called rumour routing. It uses two different types of messages that are sent around the network. The first time, 'agents' are generated at the origin node of an event. They are used to spread information about events. This is possible by maintaining routing tables to events both in sensor nodes and in agents. Each agent knows from the beginning 'his' own event but will likely learn and spread information about other events while travelling the net. Second, to obtain information, 'query' messages can be generated by specified nodes. They roam the network, trying to find a node that knows the path to the event of interest. In our implementation, this is called the 'search' mode. As soon as the 'query' messages passes a node that knows the routing, it changes to 'track' mode until arriving at the destination query that has the information about the event of interest. Then the query message backtracks to the node of origin to report about the event.

2 Compiling and Running of the Program

The program is writtne according to Java 1.7 and compiles on the commandline with standard command `javac RumourRoutingApp.java`. Invoking the compiled program is done by calling `java ./RumourRoutingApp.class` from the command line. A typical run will result in screen output similar to the following:

```
Event at x: 30 y 120, at time 1135, id 317
Event at x: 210 y 20, at time 1040, id 286
Event at x: 0 y 50, at time 1422, id 389
Event at x: 150 y 250, at time 1410, id 385
Event at x: 130 y 0, at time 1826, id 490
Event at x: 130 y 220, at time 3002, id 748
Event at x: 50 y 270, at time 4546, id 1164
Event at x: 130 y 210, at time 3848, id 977
Experiment finished
```

A number of parameters can be modfied in the `RumourRoutingApp.java` file. The meaning of these will be described in the section 'Description of Program Structure'.

```
int NODES_X                 = 50;
int NODES_Y                 = 50;
int NO_NODES                = NODES_X * NODES_Y;
double NEW_EVENTS           = 0.0001;
int NODE_RANGE              = 15;
double PROB_AGENT           = 0.5;
int TTL_AGENT               = 10;
int TTL_QUERY               = 50;
int QUERY_NODES             = 4;
int QUERY_PERIODICITY       = 400;
int TIMESTEPS               = 10000;
int NUMBER_OF_RECENT_NODES  = 5;
int QUERY_RESEND_WAIT       = TTL_QUERY *8;
```

2.1 Javadoc

JavaDoc pages were created with the built-int functions in IntelliJ and can be found in the javadoc subdirectory. The pages are in HTML format and can be viewed by opening index.html in the javadoc directory with a web browser.

2.2 Specific Design Decisions

3 Description of Program Structure

Figure 1 shows the UML diagram of the chosen design, automatically produced using IntelliJ.

4 Limitations and Future Development

5 Testing Framework

6 Individual Contributions

6.1 Johan Eklund

6.2 Tommie Lindberg

6.3 Jakob Lundin

6.4 Lorenz Gerber

References

[1] D. Braginsky and D. Estrin. Rumor routing algorthim for sensor networks. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 22–31. ACM, 2002.

[2] Umeå University, 5dv133 obligatorisk uppgift 3. `http://www8.cs.umu.se/kurser/5DV133/VT16/uppgifter/ou3/`, 2016. accessed: 2016-04-28.

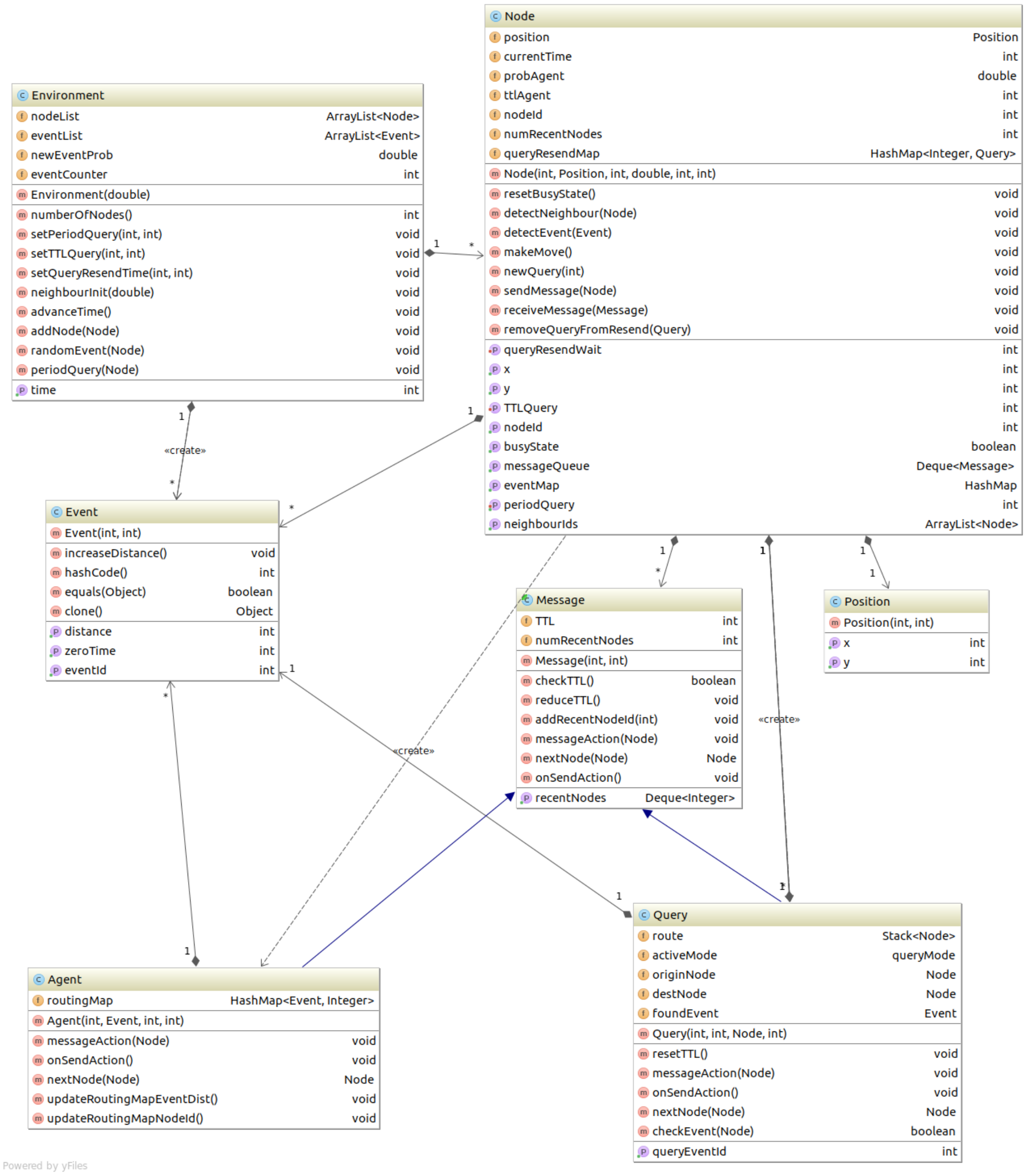

Figure 1: *UML diagram for implementing a sensory network application that allows testing of the rumour routing algoritm.*