Progetto di Simulazione di Sistemi

Samuele Evangelisti

a.a. 2019/2020

Contents

List of Figures

A Codice Sorgente

A.1 Router

```
//
// This file is part of an OMNeT++/OMNEST simulation example.
//
// Copyright (C) 2006-2015 OpenSim Ltd.
//
// This file is distributed WITHOUT ANY WARRANTY. See the file
// 'license' for details on this and other legal matters.
//

package org.omnetpp.queueing;

//
// Sends the messages to different outputs depending on a set algorithm.
// Sends the messages to first queueNumber-th queues.
//
// @author rhornig, Samuele Evangelisti
// @todo minDelay not implemented
//
simple Router
{
    parameters:
        @group(Queueing);
        @display("i=block/routing");
        string routingAlgorithm @enum("random","roundRobin","shortestQueue","minDelay","
            pssRandom") = default("random");
        volatile int randomGateIndex = default(intuniform(0, sizeof(out)-1));    // the
            destination gate in case of random routing
        // progettoss
        int queueNumber = default(sizeof(out)-1);    // queue number limit
    gates:
        input in[];
        output out[];
}
```

Listing 1: "Router.ned"

```
//
// This file is part of an OMNeT++/OMNEST simulation example.
//
// Copyright (C) 2006-2015 OpenSim Ltd.
//
// This file is distributed WITHOUT ANY WARRANTY. See the file
// 'license' for details on this and other legal matters.
//

#ifndef __QUEUEING_ROUTER_H
#define __QUEUEING_ROUTER_H

#include "QueueingDefs.h"

namespace queueing {

// routing algorithms
enum {
    ALG_RANDOM,
    ALG_ROUND_ROBIN,
    ALG_MIN_QUEUE_LENGTH,
    ALG_MIN_DELAY,
    ALG_MIN_SERVICE_TIME,
    // progettoss
    ALG_PSSRANDOM
};

/**
 * Sends the messages to different outputs depending on a set algorithm.
 * Sends the messages to first queueNumber-th queues.
 */
class QUEUEING_API Router : public cSimpleModule
{
    private:
        int routingAlgorithm;  // the algorithm we are using for routing
        int rrCounter;         // msgCounter for round robin routing
        // progettoss
        int queueNumber;
    protected:
        virtual void initialize() override;
        virtual void handleMessage(cMessage *msg) override;
};

}; //namespace

#endif
```

Listing 2: "Router.h"

```
//
// This file is part of an OMNeT++/OMNEST simulation example.
//
// Copyright (C) 2006-2015 OpenSim Ltd.
//
// This file is distributed WITHOUT ANY WARRANTY. See the file
// `license' for details on this and other legal matters.
//

#include "Router.h"

namespace queueing {

Define_Module(Router);

void Router::initialize()
{
    const char *algName = par("routingAlgorithm");
    if (strcmp(algName, "random") == 0) {
        routingAlgorithm = ALG_RANDOM;
    }
    else if (strcmp(algName, "roundRobin") == 0) {
        routingAlgorithm = ALG_ROUND_ROBIN;
    }
    else if (strcmp(algName, "minQueueLength") == 0) {
        routingAlgorithm = ALG_MIN_QUEUE_LENGTH;
    }
    else if (strcmp(algName, "minDelay") == 0) {
        routingAlgorithm = ALG_MIN_DELAY;
    }
    else if (strcmp(algName, "minServiceTime") == 0) {
        routingAlgorithm = ALG_MIN_SERVICE_TIME;
    }
    else if (strcmp(algName, "pssRandom") == 0) {
        routingAlgorithm = ALG_PSSRANDOM;
    }
    rrCounter = 0;
    int qn = par("queueNumber").intValue() - 1;
    if (qn < 0 || qn > gateSize("out") - 1)
        throw cRuntimeError("Invalid queue number");
    else
        queueNumber = qn;
}

void Router::handleMessage(cMessage *msg)
{
    int outGateIndex = -1;  // by default we drop the message

    switch (routingAlgorithm) {
        case ALG_RANDOM:
            outGateIndex = par("randomGateIndex");
            break;

        case ALG_ROUND_ROBIN:
            outGateIndex = rrCounter;
            rrCounter = (rrCounter + 1) % gateSize("out");
            break;

        case ALG_MIN_QUEUE_LENGTH:
            // TODO implementation missing
            outGateIndex = -1;
            break;

        case ALG_MIN_DELAY:
            // TODO implementation missing
            outGateIndex = -1;
            break;

        case ALG_MIN_SERVICE_TIME:
            // TODO implementation missing
            outGateIndex = -1;
            break;

        case ALG_PSSRANDOM:
            outGateIndex = intuniform(0, queueNumber);
            break;

        default:
            outGateIndex = -1;
            break;
    }

    // send out if the index is legal
    if (outGateIndex < 0 || outGateIndex >= gateSize("out"))
        throw cRuntimeError("Invalid output gate selected during routing");

    send(msg, "out", outGateIndex);
}
```

```

}; //namespace
```

Listing 3: "Router.cc"