**Лабораторная работа №6. Изучение системы имитационного моделирования OMNET++**

*Цель работы.*Изучение основ работы с системой имитационного моделирования OMNET++, построение простейшей модели и исследование ее функционирования.

Исходные данные и требования (таблица 6.2)

Таблица 6.2

Характеристики и параметры СМО

|  |  |  |
| --- | --- | --- |
| **Параметр** | **Ед.изм.** | **Значение** |
| Количество узлов в сети | шт. | 5 |
| Номер узла удаления пакета | - | 3 |
| Задержка в линии | с | 0,12 |
| Структура сети | - | Произвольная полносвязная, связность  узлов от 1 до 5. |

Структура модели (рис. 6.18).

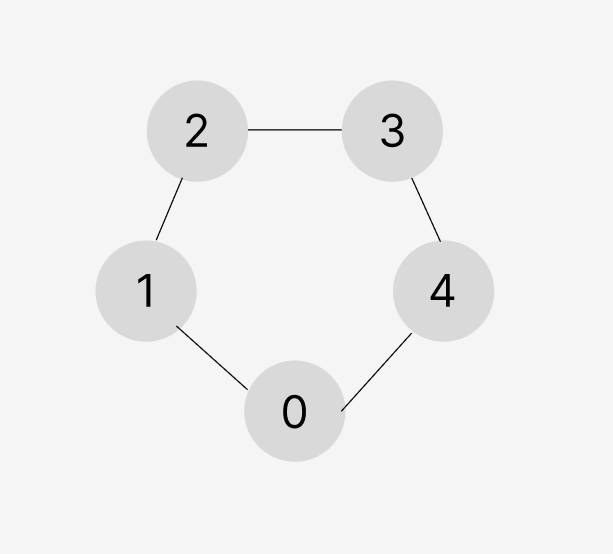


Рис. 6.18 – Структура имитационной модели

6.3.1 Построение имитационной модели

Имитационная модель включает в себя один тип узла: «tic». Все узлы создаются на основе этого типа. На этапе инициализации узел «0» создает сообщение и отправляет узлам на случайный выход, т. е. к узлам 1 или 4. Далее узлы «1» или «4» пересылают принятые сообщения в случайных направлениях и так процесс повторяется, пока сообщение не достигнет узла с заданным номером (номер 3). При приеме сообщения этим узлом,Ю оно удаляется и процесс моделирования завершается.

Структура модели, построенной в OMNeT++, приведена на рис. 6.19.

Файл txc10.cc

**#include** <stdio.h>

**#include** <string.h>

**#include** <omnetpp.h>

**class** Txc10 : **public** omnetpp::cSimpleModule

{

**protected**:

**virtual** **void** **forwardMessage**(omnetpp::cMessage \*msg); **virtual** **void** **initialize**();

**virtual** **void** **handleMessage**(omnetpp::cMessage \*msg);

};

Define\_Module(Txc10); **void** **Txc10::initialize**()

{

**if** (**getIndex**()==0)

{

// Если номер узла 0, то создать сообщение.

**char** msgname[20];

sprintf(msgname, "tic-%d", **getIndex**()); omnetpp::cMessage \*msg = **new** omnetpp::cMessage(msgname); **scheduleAt**(0.0, msg); //Отправить сообщение

}

}

**void** **Txc10::handleMessage**(omnetpp::cMessage \*msg)

{

**if** (**getIndex**()==2) //Если номер узла 3, то удалить сообщение

{

// Обработка поступившего сообщения.

EV << "Message " << msg << " arrived.\n"; **delete** msg;

}

**else**

{

// Иначе переслать сообщение далее.

forwardMessage(msg);

}

}

**void** **Txc10::forwardMessage**(omnetpp::cMessage \*msg)

{

// Пересылка сообщения.

// На случайный порт `out[]'.

**int** n = **gateSize**("out"); **int** k = intuniform(0,n-1);

EV << "Forwarding message " << msg << " on port out[" << k << "]\n"; send(msg, "out", k);

}

Файл tictoc10.ned

*//*

*// This file is part of an OMNeT++/OMNEST simulation example.*

*// Copyright (C) 2003-2008 Andras Varga*

*//*

*// This file is distributed WITHOUT ANY WARRANTY. See the file*

*// `license' for details on this and other legal matters.*

*//*

**simple**Txc10 *{*

**parameters***:*

**@display***(*"i=block/routing"*);*

**gates***:*

**input**in*[]; // declare in[] and out[] to be vector gates*

**output**out*[];*

*}*

**network**Tictoc10 *{*

**submodules***:*

tic*[*5*]:* Txc10*; // Теперь только 5 модулей (индексы от 0 до 4)*

**connections***:*

*// Соединения для кольцевой топологии*

tic*[*0*].*out*++* **-->** *{* delay *=* 100ms*; }* **-->**tic*[*1*].*in*++;*

tic*[*0*].*in*++* **<--** *{* delay *=* 100ms*; }* **<--**tic*[*1*].*out*++;*

tic*[*1*].*out*++* **-->** *{* delay *=* 100ms*; }* **-->**tic*[*2*].*in*++;*

tic*[*1*].*in*++* **<--** *{* delay *=* 100ms*; }* **<--**tic*[*2*].*out*++;*

tic*[*2*].*out*++* **-->** *{* delay *=* 100ms*; }* **-->**tic*[*3*].*in*++;*

tic*[*2*].*in*++* **<--** *{* delay *=* 100ms*; }* **<--**tic*[*3*].*out*++;*

tic*[*3*].*out*++* **-->** *{* delay *=* 100ms*; }* **-->**tic*[*4*].*in*++;*

tic*[*3*].*in*++* **<--** *{* delay *=* 100ms*; }* **<--**tic*[*4*].*out*++;*

*// Замыкаем кольцо: последний модуль соединяем с первым*

tic*[*4*].*out*++* **-->** *{* delay *=* 100ms*; }* **-->**tic*[*0*].*in*++;*

tic*[*4*].*in*++* **<--** *{* delay *=* 100ms*; }* **<--**tic*[*0*].*out*++;*

*}*

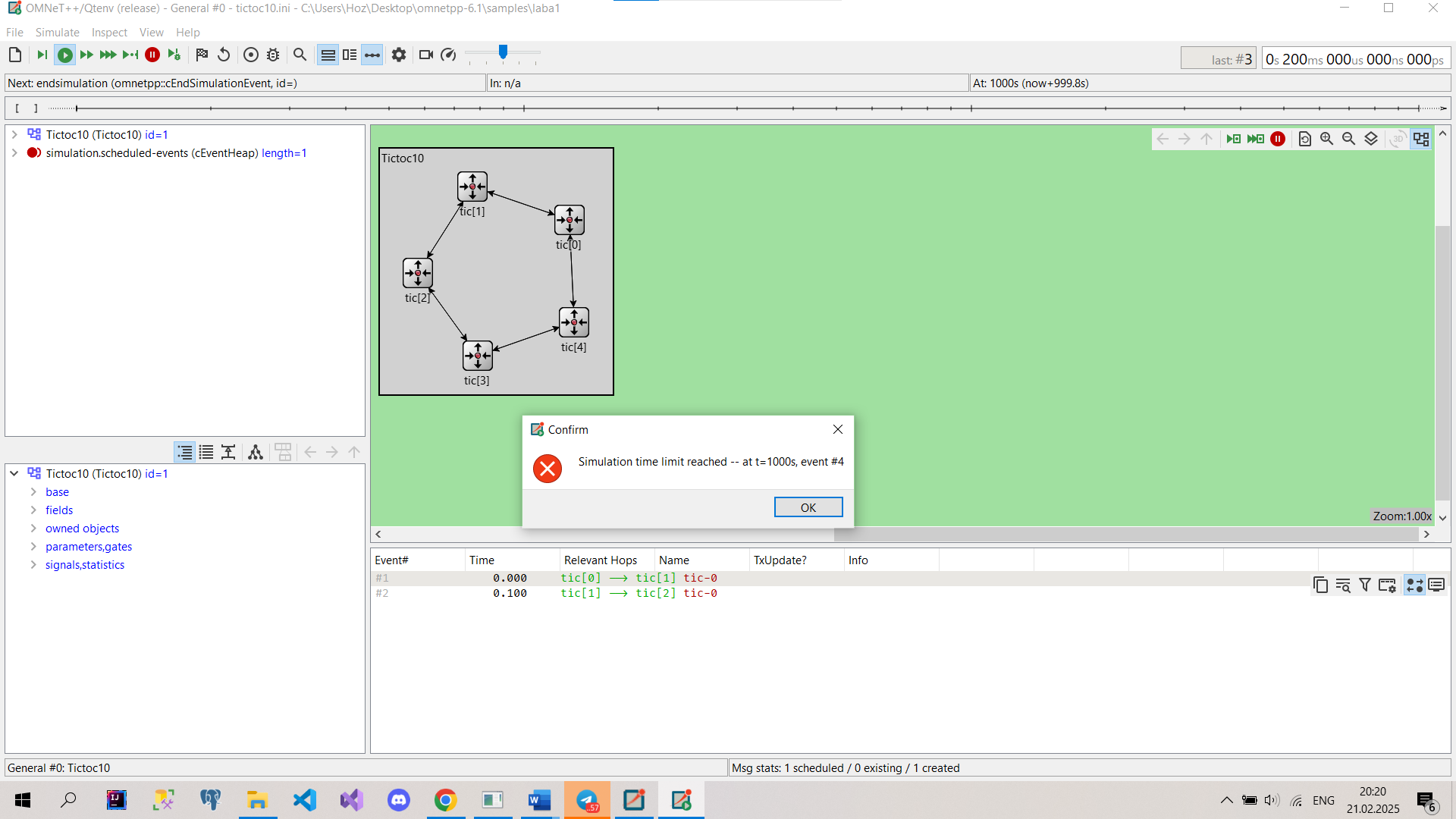


Рис. 6.19 – Имитационная модель (структура сети)

На рис. 6.19 приведена структура имитационной модели сети из 11 узлов, структура которой эквивалентна рис. 6.18.

6.3.2 Выполнение имитационного эксперимента (результаты имитационного моделирования)

На рис. 6.20 приведены результаты выполнения имитационного эксперимента.

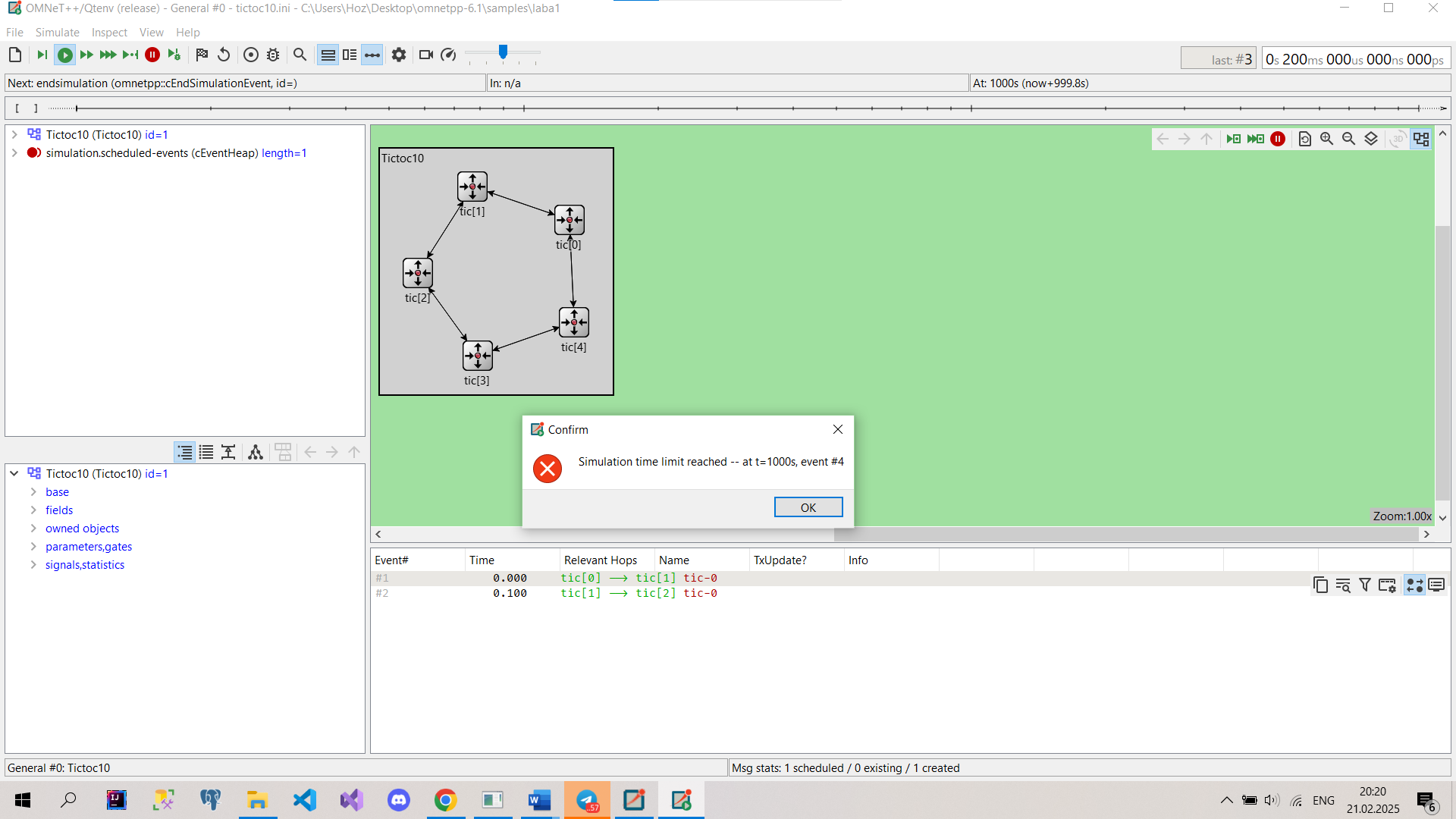


Рис. 6.20 – Имитационная модель (структура сети)

Имитационная модель завершает работу, когда сообщение поступает в узел с номером 2, что видно из лог-листинга, приведенного на рис. 6.20.

Маршрут, пройденный сообщением:

0, 1, 2

6.3.3 Выводы

1. Для построения имитационной модели сети была разработана модель простого модуля на языке C++. Для построения моделей всех узлов сети используется один и тот же простой модуль.

2. Функционирование простого модуля определяется условиями, которые учитывают его место в структуре сети. Если номер модуля равен 0, то он является генератором сообщения, если его номер равен заданному в задании (в данном случае 5), то он является конечным узлом маршрута.

3. Выбор маршрута для доставки сообщения производится случайным образом, при приеме сообщения узлом он вырабатывает случайное число, на основе которого выбирает порт назначения.

