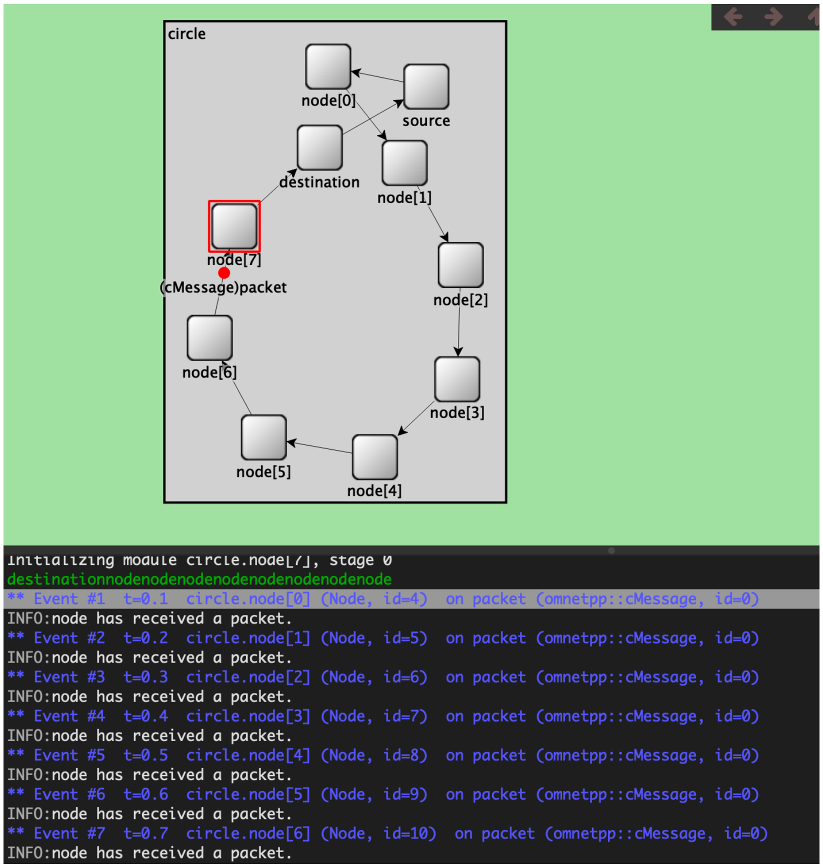
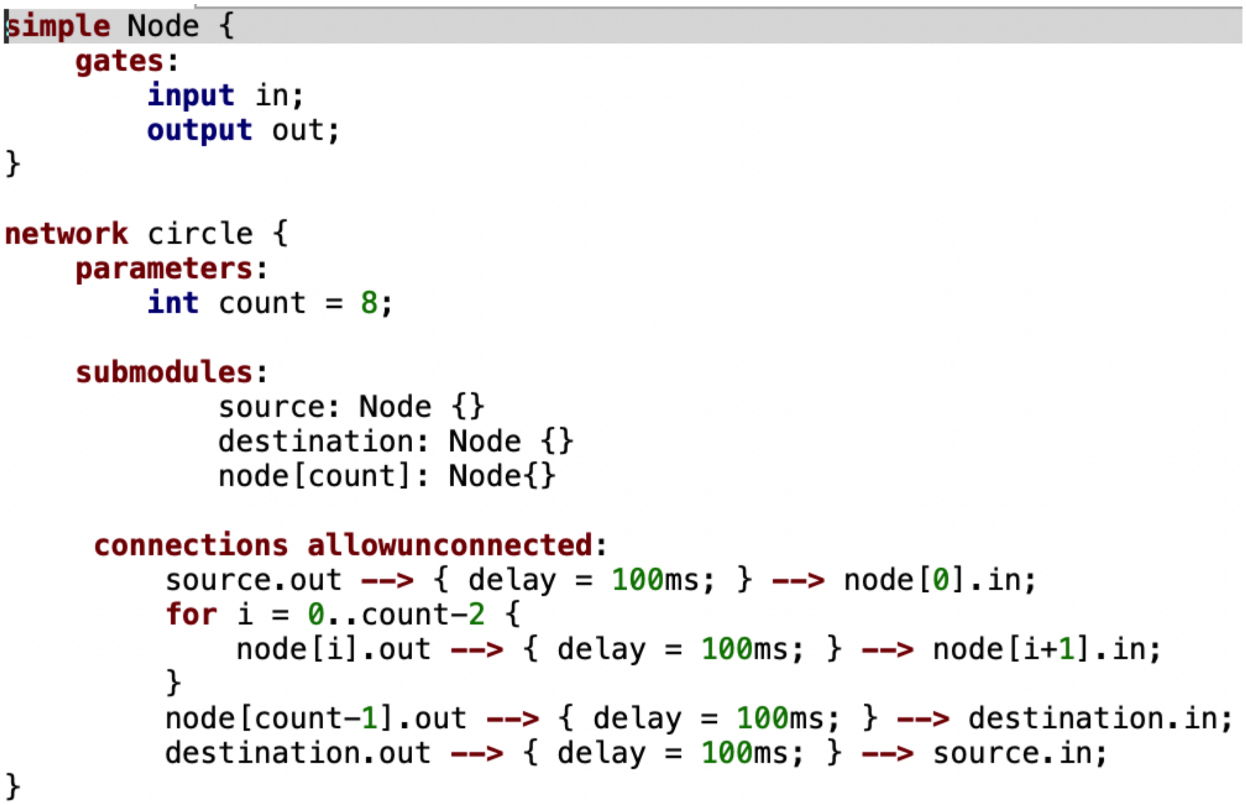


```
initializing module circle.node[7], stage 0
destinationnodenodenodenodenodenodenodenode
** Event #1  t=0.1  circle.node[0] (Node, id=4)  on packet (omnetpp::cMessage, id=0)
INFO:node has received a packet.
** Event #2  t=0.2  circle.node[1] (Node, id=5)  on packet (omnetpp::cMessage, id=0)
INFO:node has received a packet.
** Event #3  t=0.3  circle.node[2] (Node, id=6)  on packet (omnetpp::cMessage, id=0)
INFO:node has received a packet.
** Event #4  t=0.4  circle.node[3] (Node, id=7)  on packet (omnetpp::cMessage, id=0)
INFO:node has received a packet.
** Event #5  t=0.5  circle.node[4] (Node, id=8)  on packet (omnetpp::cMessage, id=0)
INFO:node has received a packet.
** Event #6  t=0.6  circle.node[5] (Node, id=9)  on packet (omnetpp::cMessage, id=0)
INFO:node has received a packet.
** Event #7  t=0.7  circle.node[6] (Node, id=10)  on packet (omnetpp::cMessage, id=0)
INFO:node has received a packet.
```

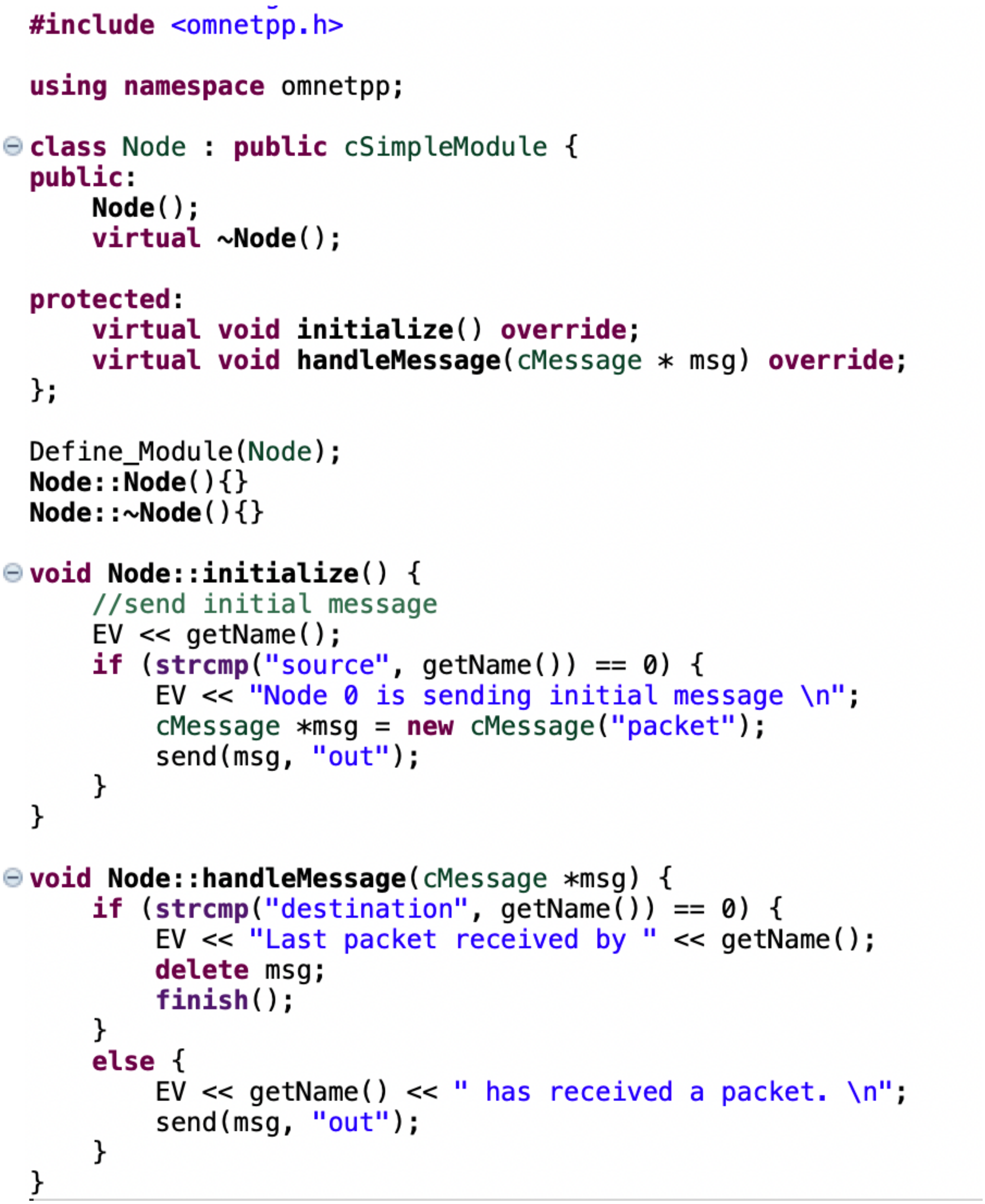


```
simple Node {
    gates:
        input in;
        output out;
}

network circle {
    parameters:
        int count = 8;

    submodules:
            source: Node {}
            destination: Node {}
            node[count]: Node{}

     connections allowunconnected:
        source.out --> { delay = 100ms; } --> node[0].in;
        for i = 0..count-2 {
            node[i].out --> { delay = 100ms; } --> node[i+1].in;
        }
        node[count-1].out --> { delay = 100ms; } --> destination.in;
        destination.out --> { delay = 100ms; } --> source.in;
}
```



```cpp
#include <omnetpp.h>

using namespace omnetpp;

class Node : public cSimpleModule {
public:
    Node();
    virtual ~Node();

protected:
    virtual void initialize() override;
    virtual void handleMessage(cMessage * msg) override;
};

Define_Module(Node);
Node::Node(){}
Node::~Node(){}

void Node::initialize() {
    //send initial message
    EV << getName();
    if (strcmp("source", getName()) == 0) {
        EV << "Node 0 is sending initial message \n";
        cMessage *msg = new cMessage("packet");
        send(msg, "out");
    }
}

void Node::handleMessage(cMessage *msg) {
    if (strcmp("destination", getName()) == 0) {
        EV << "Last packet received by " << getName();
        delete msg;
        finish();
    }
    else {
        EV << getName() << " has received a packet. \n";
        send(msg, "out");
    }
}
```