Отчёт по лабораторной работе №1

Дисциплина: Имитационное моделирование

Ганина Таисия Сергеевна, НФИбд-01-22

Содержание

# 1 Цель работы

Целью данной работы является приобретение навыков моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также анализ полученных результатов моделирования.

# 2 Задание

1. Записать шаблон сценария для NS-2.
2. Создать простой пример описания топологии сети, состоящей из двух узлов и одного соединения.
3. Создать пример с усложнённой топологией сети.
4. Создать пример с кольцевой топологией сети.
5. Выполнить упражнение.

# 3 Теоретическое введение

Network Simulator (NS-2) — один из программных симуляторов моделирования процессов в компьютерных сетях. NS-2 позволяет описать топологию сети, конфигурацию источников и приёмников трафика, параметры соединений (полосу пропускания, задержку, вероятность потерь пакетов и т.д.) и множество других параметров моделируемой системы. Данные о динамике трафика, состоянии соединений и объектов сети, а также информация о работе протоколов фиксируютсяв генерируемом trace-файле.

NS-2 является объектно-ориентированным программным обеспечением. Его ядро реализовано на языке С++. В качестве интерпретатора используется язык скриптов (сценариев) OTcl (Object oriented Tool Command Language). NS-2 полностью поддерживает иерархию классов С++ и подобную иерархию классов интерпретатора OTcl.

Обе иерархии обладают идентичной структурой, т.е. существует однозначное соответствие между классом одной иерархии и таким же классом другой. Объединение для совместного функционирования С++ и OTcl производится при помощи TclCl (Classes Tcl). В случае, если необходимо реализовать какую-либо специфическую функцию, не реализованную в NS-2 на уровне ядра, для этого используется код на С++.

**Процесс создания модели сети для NS-2 состоит из нескольких этапов:**

1. Создание нового объекта класса Simulator, в котором содержатся методы, необходимые для дальнейшего описания модели (например, методы new и delete используются для создания и уничтожения объектов соответственно);
2. Описание топологии моделируемой сети с помощью трёх основных функциональных блоков: узлов (nodes), соединений (links) и агентов (agents);
3. Задание различных действий, характеризующих работу сети.

Для создания узла используется метод node. При этом каждому узлу автоматически присваивается уникальный адрес. Для построения однонаправленных и двунаправленных линий соединения узлов используют методы simplex-link и duplex-link соответственно.

Важным объектом NS-2 являются агенты, которые могут рассматриваться как процессы и/или как транспортные единицы, работающие на узлах моделируемой сети. Агенты могут выступать в качестве источников трафика или приёмников, а также как динамические маршрутизирующие и протокольные модули. Агенты создаются с помощью методов общего класса Agent и являются объектами его подкласса, т.е. Agent/type, где type определяет тип конкретного объекта. Например, TCP-агент может быть создан с помощью команды: set tcp [ new Agent/TCP ]

Для закрепления агента за конкретным узлом используется метод attach-agent. Каждому агенту присваивается уникальный адрес порта для заданного узла (аналогично портам tcp и udp). Чтобы за конкретным агентом закрепить источник, используют методы attach-source и attach-traffic. Например, можно прикрепить ftp или telnet источники к TCP-агенту. Есть агенты, которые генерируют свои собственные данные, например, CBR-агент (Constant Bit-Rate) — источник трафика с постоянной интенсивностью. Действия разных агентов могут быть назначены планировщиком событий (Event Scheduler) в определённые моменты времени (также в определённые моменты времени могут быть задействованы или отключены те или иные источники данных, запись статистики, разрыв, либо восстановление соединений, реконфигурация топологии и т.д.).

Для этого может использоваться метод at. Моделирование начинается при помощи метода run. В качестве дополнения к NS-2 часто используют средство визуализации nam (network animator) для графического отображения свойств моделируемой системы и проходящего через неё трафика и пакет Xgraph для графического представления результатов моделирования. Запуск сценария NS-2 осуществляется в командной строке с помощью команды: ns [tclscript]

Здесь [tclscript] — имя файла скрипта Tcl, который определяет сценарий моделирования (т.е. топологию и различные события). Nam можно запустить с помощью команды nam [nam-file]

Здесь [nam-file] — имя nam trace-файла, сгенерированного с помощью ns.

# 4 Выполнение лабораторной работы

1. Я создала в рабочем каталоге директорию mip, к которой будут выполняться лабораторные работы. Внутри mip создала директорию lab-ns, а в ней файл shablon.tcl. Открыла файл шаблона на редактирование и заполнила его согласно коду из задания лабораторной. После этого запустила шаблон и посмотрела на результат. Пустой экран приложения, где, редактируя шаблон, в будущем мы будем добавлять сети определенных топологий. (рис. [1](#fig:001), [2](#fig:002), [3](#fig:003)).

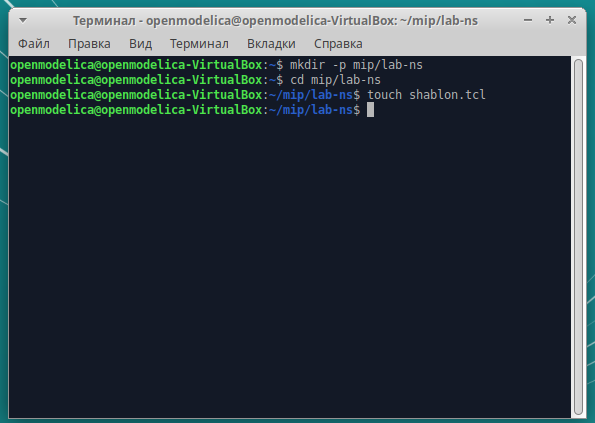


Figure 1: Создание файлов

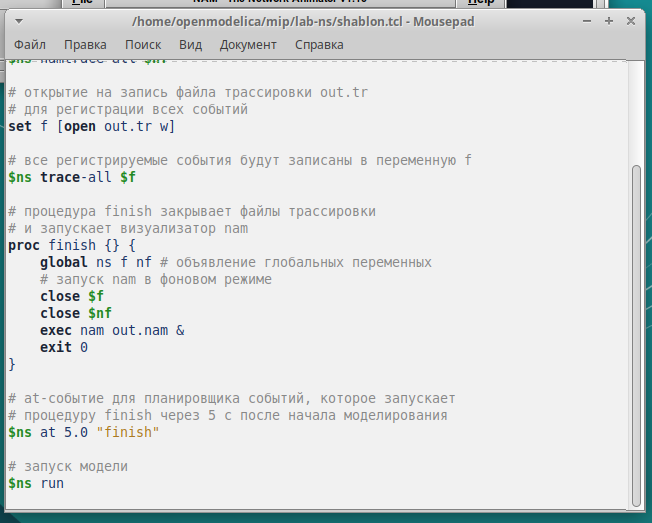


Figure 2: Заполнение файла шаблона

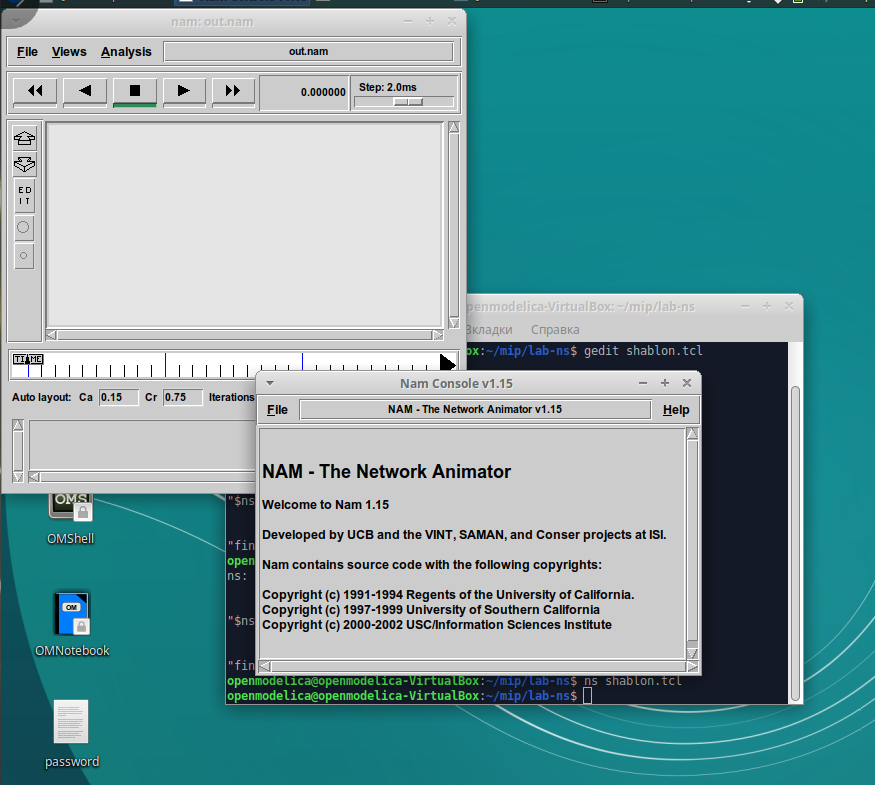


Figure 3: Результат выполнения файла шаблона, приложение

1. Я выполнила по образцу из задания лабораторной простой пример описания топологии сети, состоящей из двух узлов и одного соединения. В начале скопировала содержимое шаблона в новый файл, после отредактировала его. Добавила описание топологии сети, создала агенты для генерации и приема трафика, создала агент UDP и присоединила к узлу n0. В узле агент сам не может генерировать трафик, он лишь реализует протоколы и алгоритмы транспортного уровня. Поэтому к агенту присоединяется приложение. В данном случае — это источник с постоянной скоростью (Constant Bit Rate, CBR), который каждые 5 мс посылает пакет R = 500 байт. (рис. [4](#fig:004), [5](#fig:005), [6](#fig:006), [7](#fig:007), [8](#fig:008)).

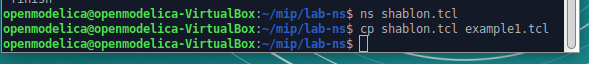


Figure 4: Копирование шаблона

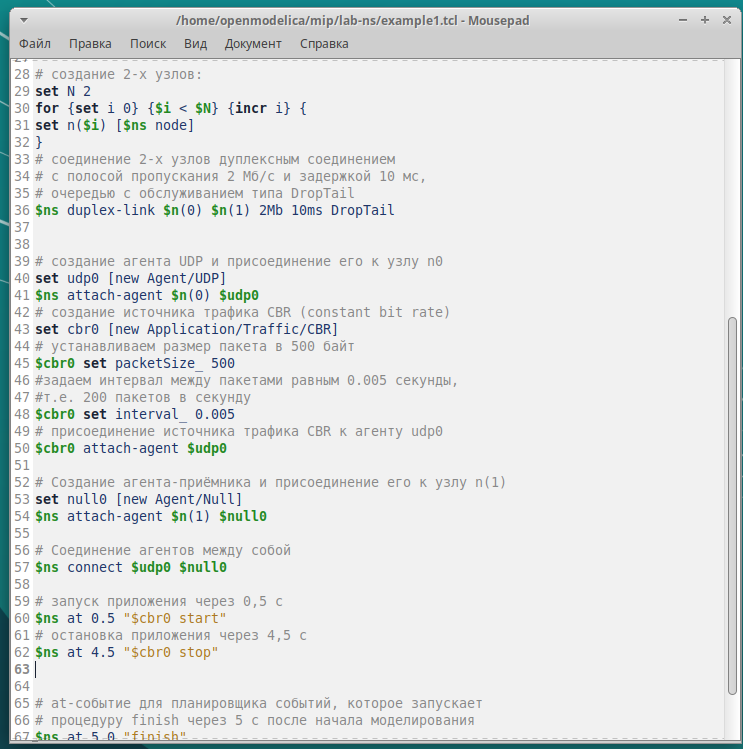


Figure 5: Код программы

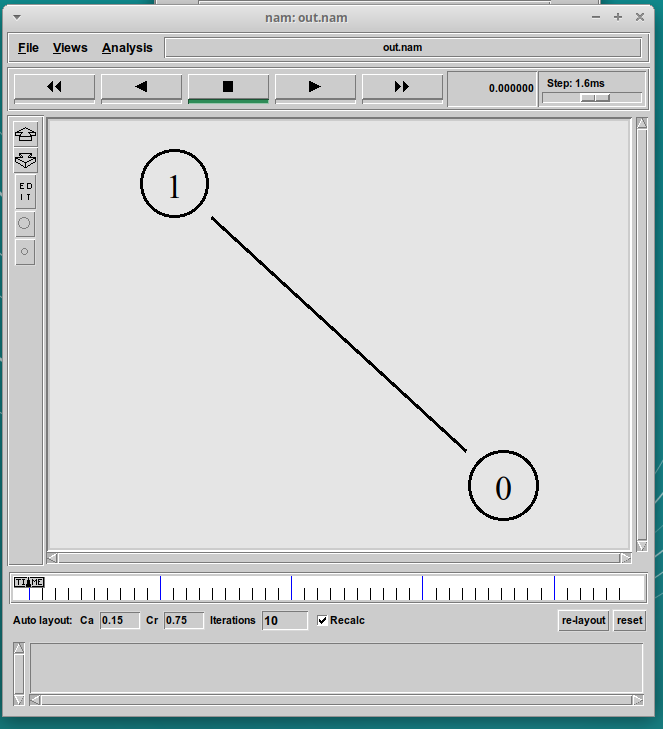


Figure 6: Аниматор nam

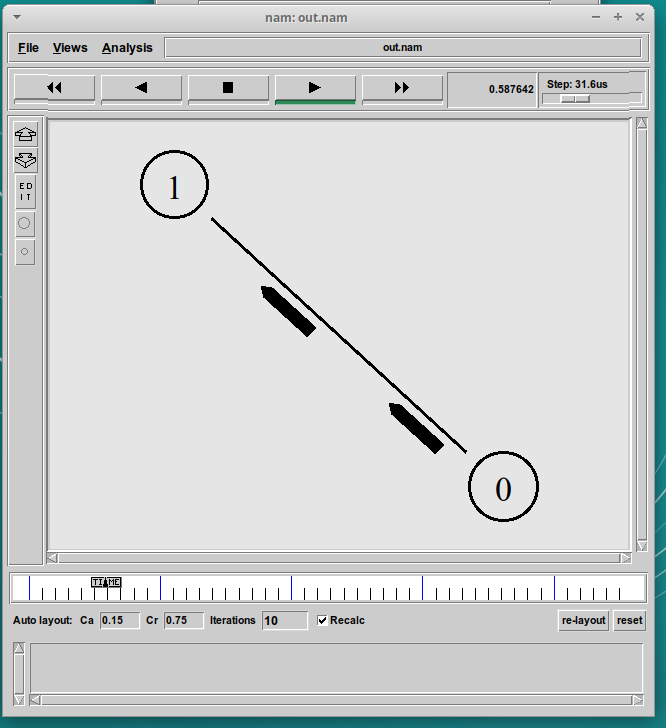


Figure 7: Визуализация простой модели сети с помощью nam

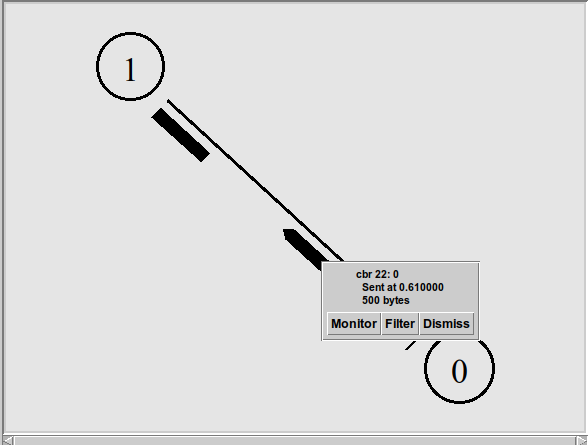


Figure 8: Можно осуществлять наблюдение за отдельным пакетом, щёлкнув по нему в окне nam

1. Пример с усложнённой топологией сети. Точно также скопировала шаблон и заполнила его, прочитав постановку задачи. Обнаружила неточности в коде, который предлагается в качестве примера в лабораторной работе, исправила их.

* Задание: “– между узлами n2 и n3 установлено дуплексное соединение с пропускной способностью 1,7 Мбит/с и задержкой 20 мс;”
* Код: $ns duplex-link $n(3) $n(2) 2Mb 10ms DropTail.
* Заменила 2 на 1.7 и 10 на 20.

Далее: - Задание: “работа cbr начинается в 0,1 секунду и прекращается в 4,5 секунды, а ftp начинает работать в 1,0 секунду и прекращает в 4,0 секунды.” - Код: $ns at 0.5 "$cbr0 start". - Заменила 0.5 на 0.1.

Сохранив изменения в отредактированном файле и запустив симулятор, получила анимированный результат моделирования. (рис. [9](#fig:009), [10](#fig:010), [11](#fig:011), [12](#fig:012), [13](#fig:013)).

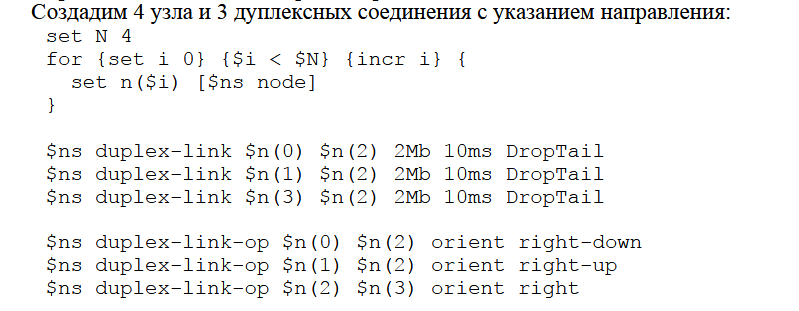


Figure 9: Кадр с неточностью

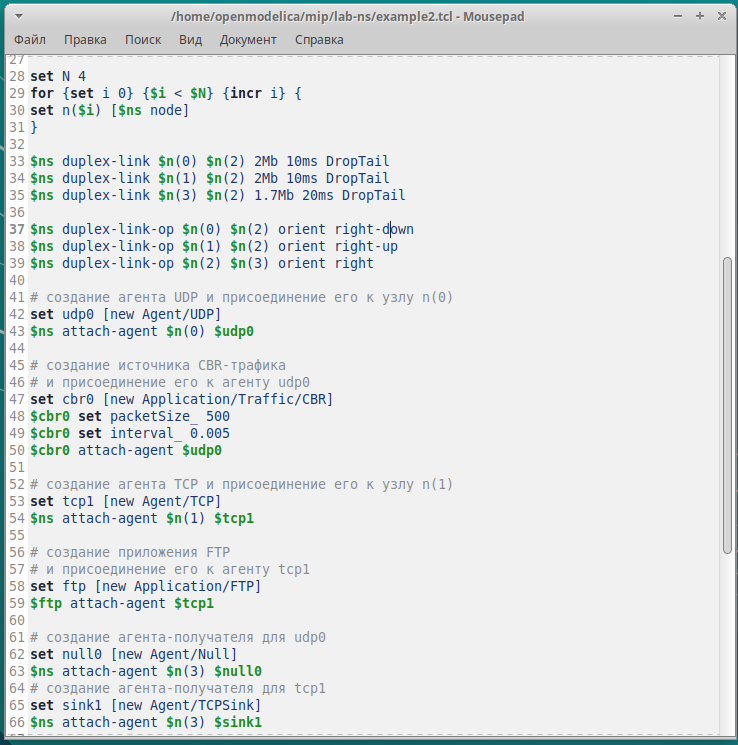


Figure 10: Код программы

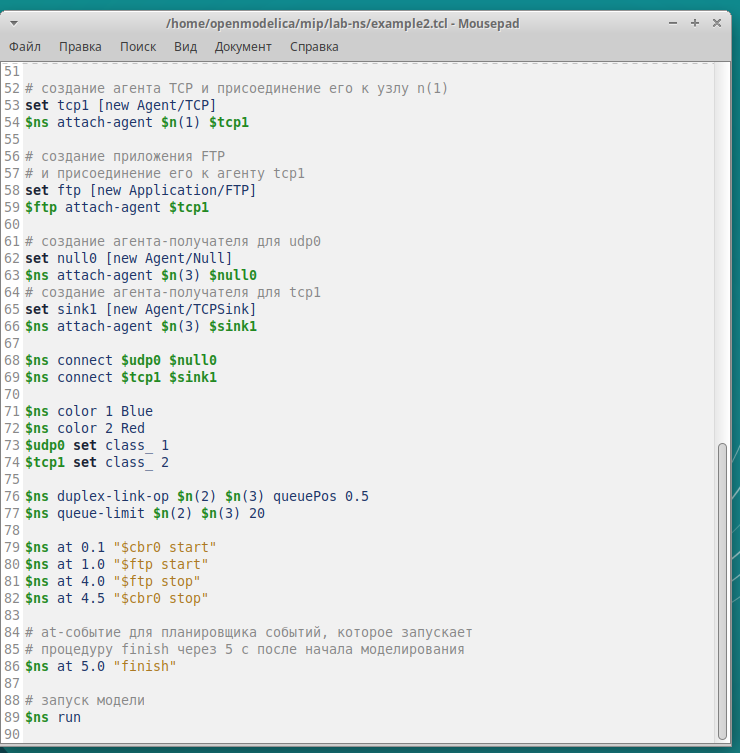


Figure 11: Продолжение кода

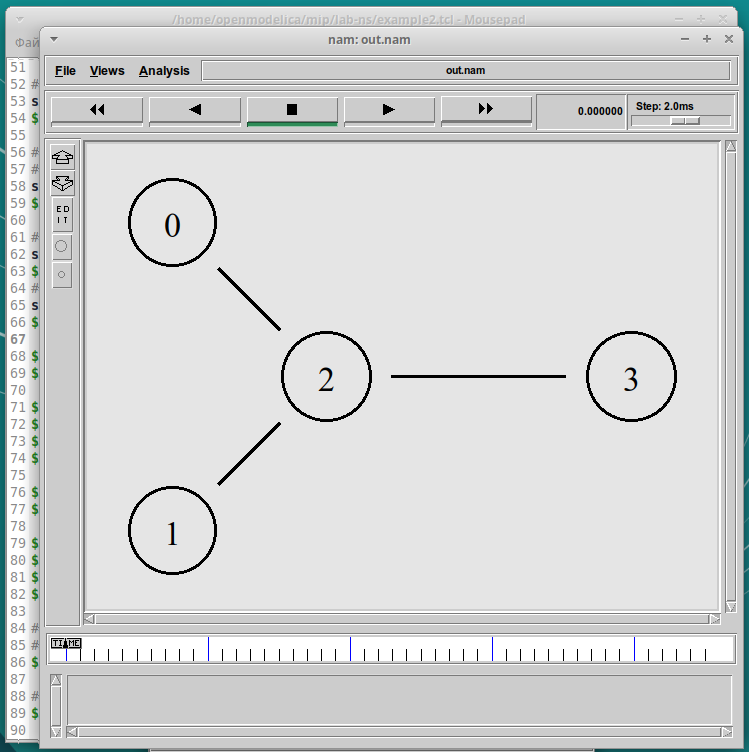


Figure 12: До запуска анимации, вид на топологию

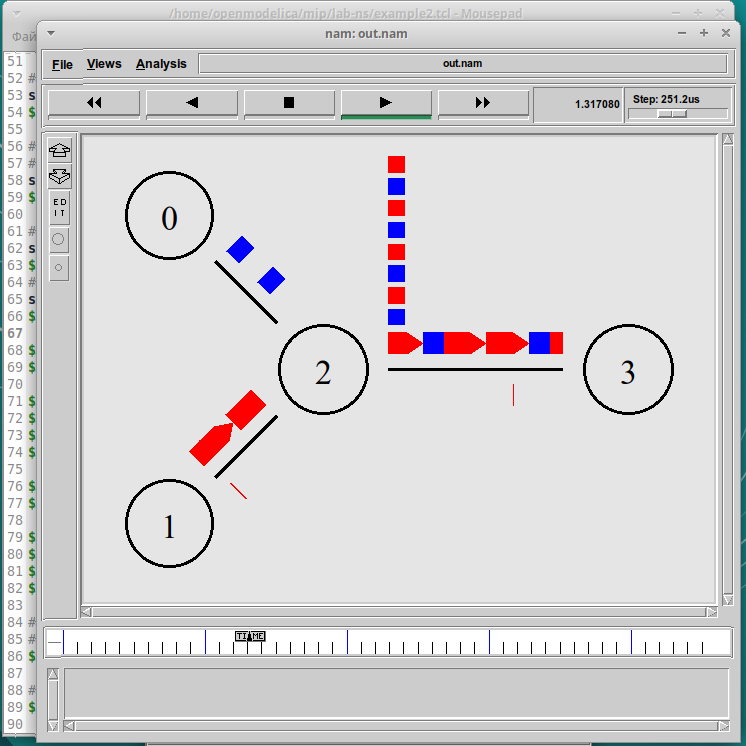


Figure 13: Выполнение моделирование, анимация

1. Пример с кольцевой топологией сети. Я опять переписала код, в данном примере данные должны передаваться по кратчайшему маршруту, а при разрыве соединения информация о топологии должна обновляться и пакеты отсылаться по новому маршруту. После восстановления соединения снова будет выбран самый короткий маршрут. (рис. [14](#fig:014), [15](#fig:015), [16](#fig:016), [17](#fig:017), [18](#fig:018)).

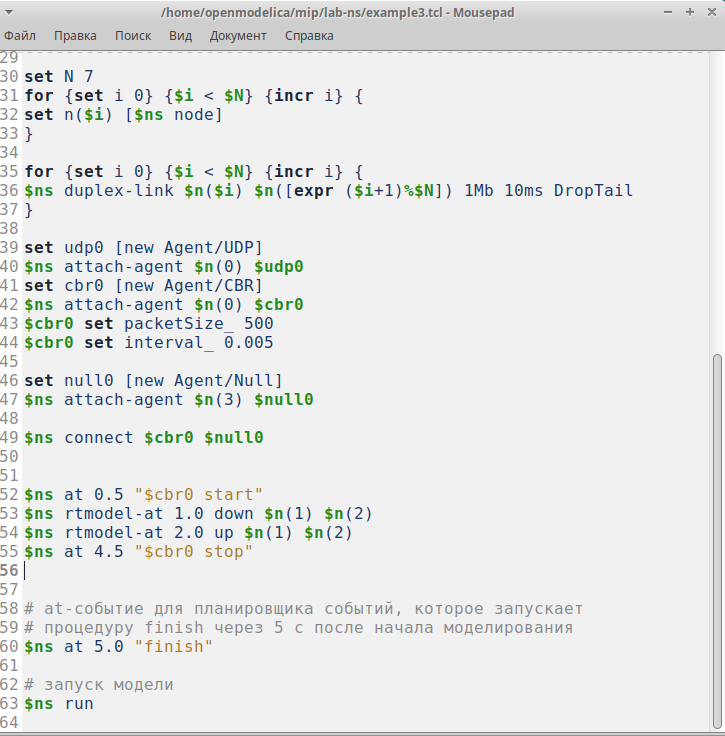


Figure 14: Текст скрипта, измененный шаблон

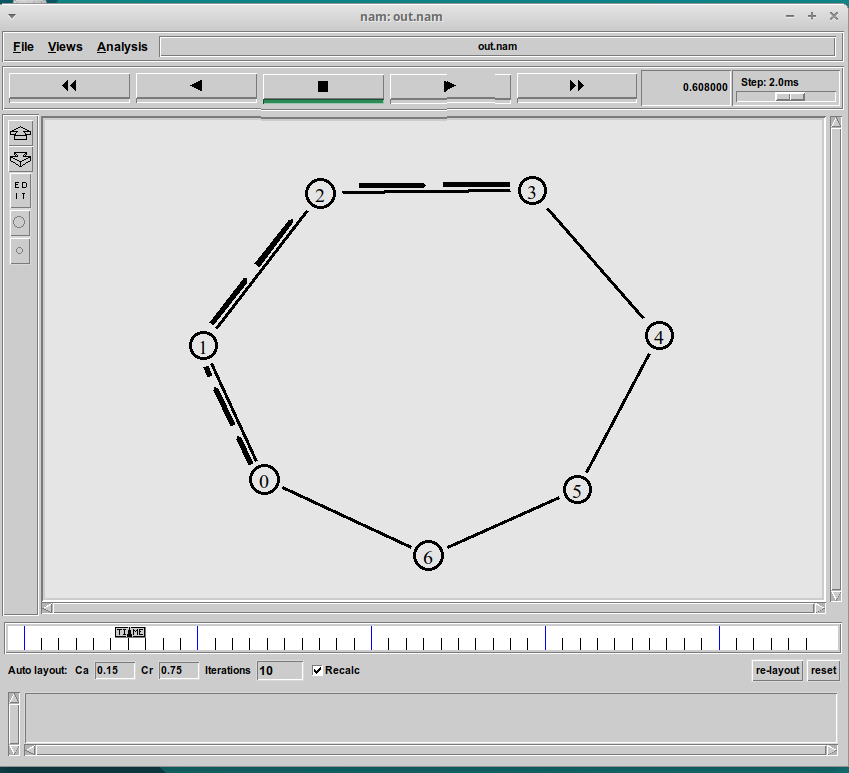


Figure 15: Кратчайший маршрут до разрыва соединения

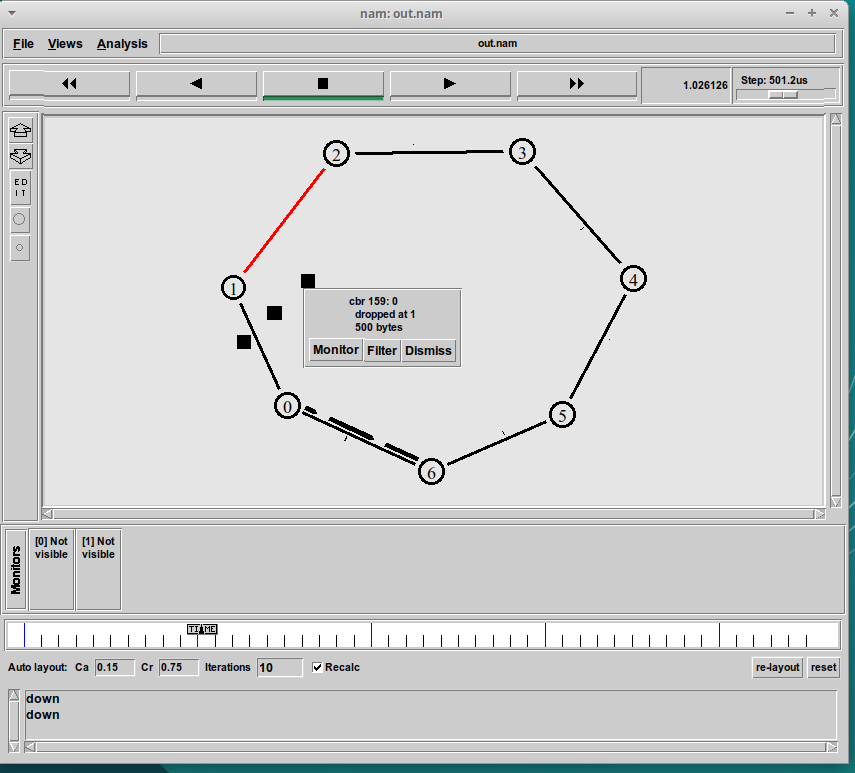


Figure 16: Момент разрыва соединения

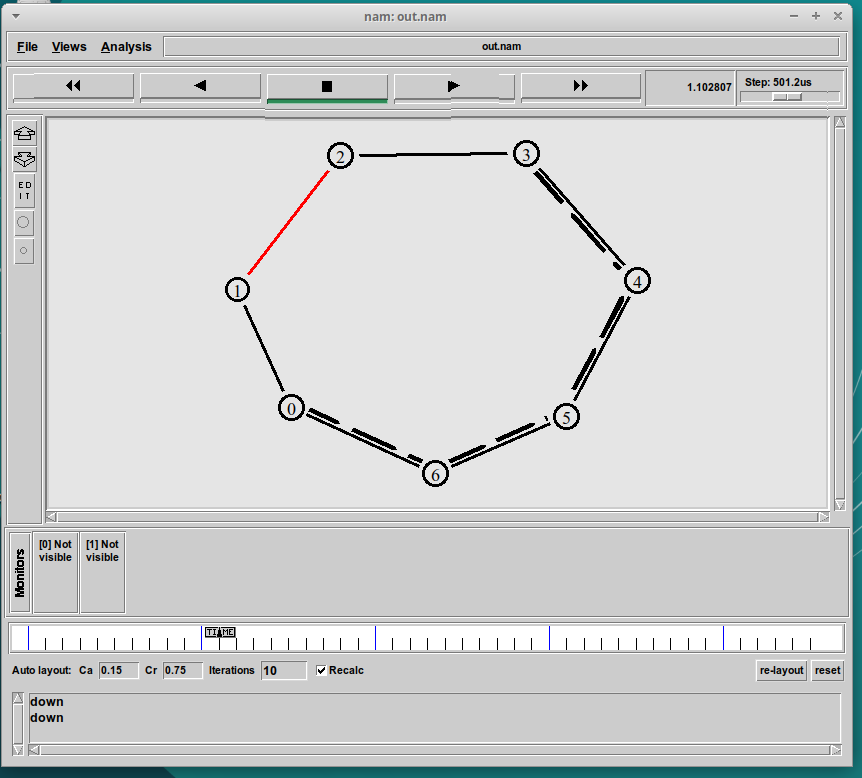


Figure 17: Резервный маршрут через узлы n(6), n(5) и n(4)

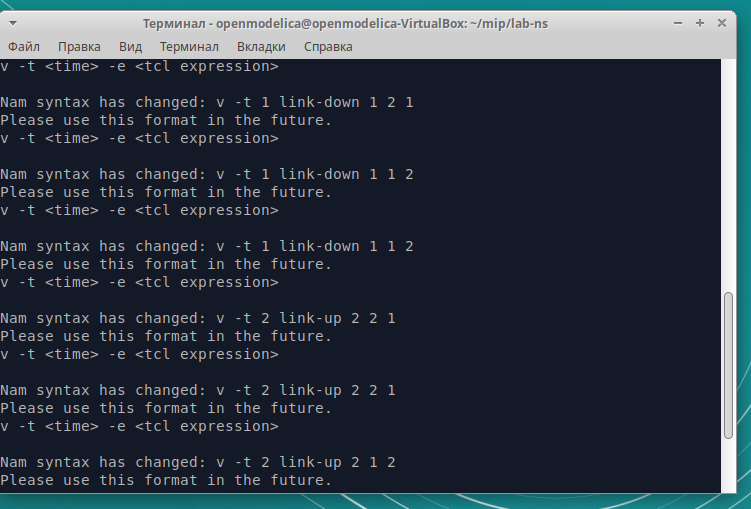


Figure 18: Сообщения в терминале о разрыве и восстановлении

1. Упражнение. Я внесла следующие изменения в реализацию примера с кольцевой топологией сети:

* топология сети должна соответствовать представленной в лабораторной.
* передача данных должна осуществляться от узла n(0) до узла n(5) по кратчайшему пути в течение 5 секунд модельного времени;
* передача данных должна идти по протоколу TCP (тип Newreno), на принимающей стороне используется TCPSink-объект типа DelAck; поверх TCP работает протокол FTP с 0,5 до 4,5 секунд модельного времени;
* с 1 по 2 секунду модельного времени происходит разрыв соединения между узлами n(0) и n(1);
* при разрыве соединения маршрут передачи данных должен измениться на резервный, после восстановления соединения пакеты снова должны пойти по кратчайшему пути. (рис. [19](#fig:019), [20](#fig:020), [21](#fig:021), [22](#fig:022), [23](#fig:023), [24](#fig:024)).

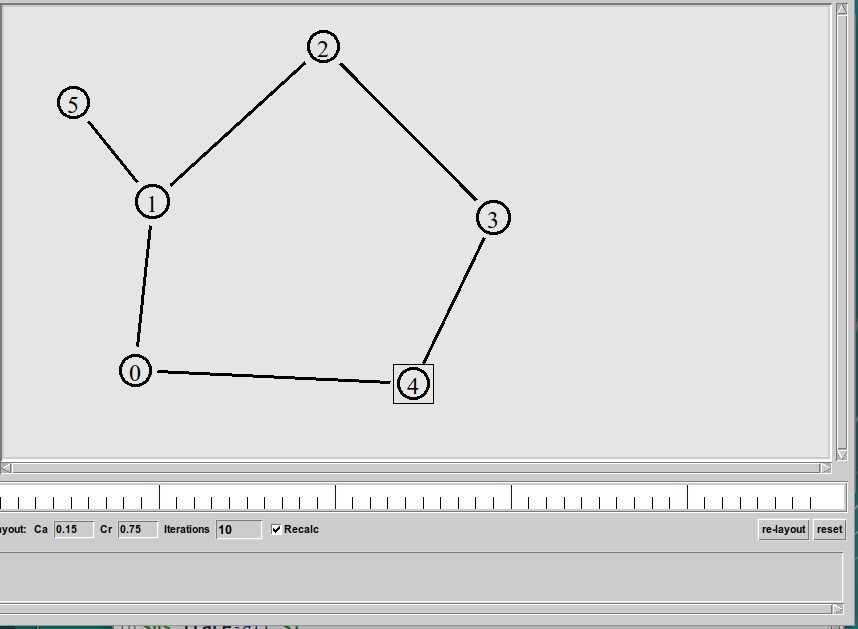


Figure 19: Топология сети

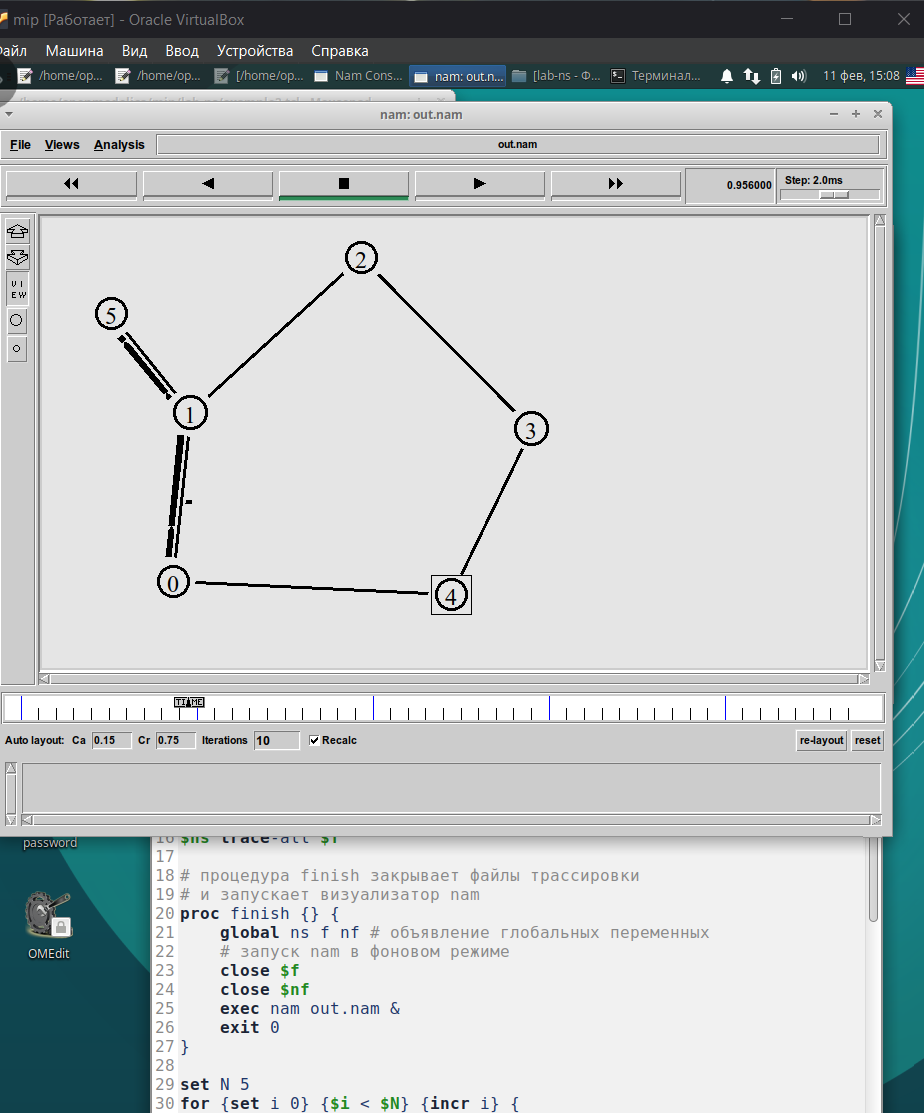


Figure 20: Передача данных до разрыва соединения между 0 и 1 узлами

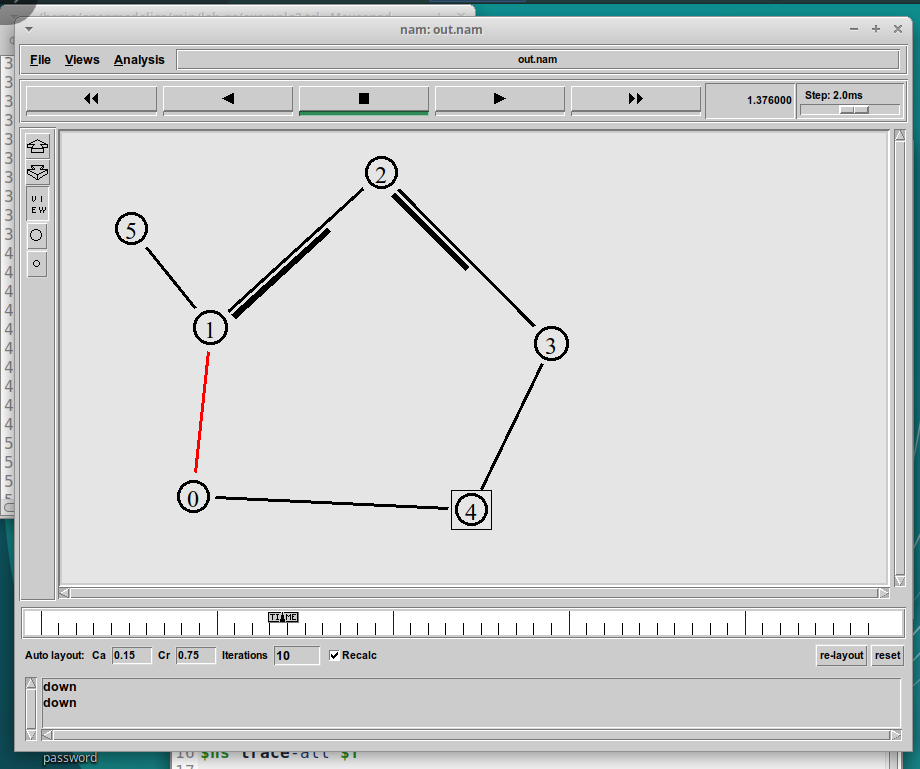


Figure 21: Передача после разрыва соединения и обновления маршрута

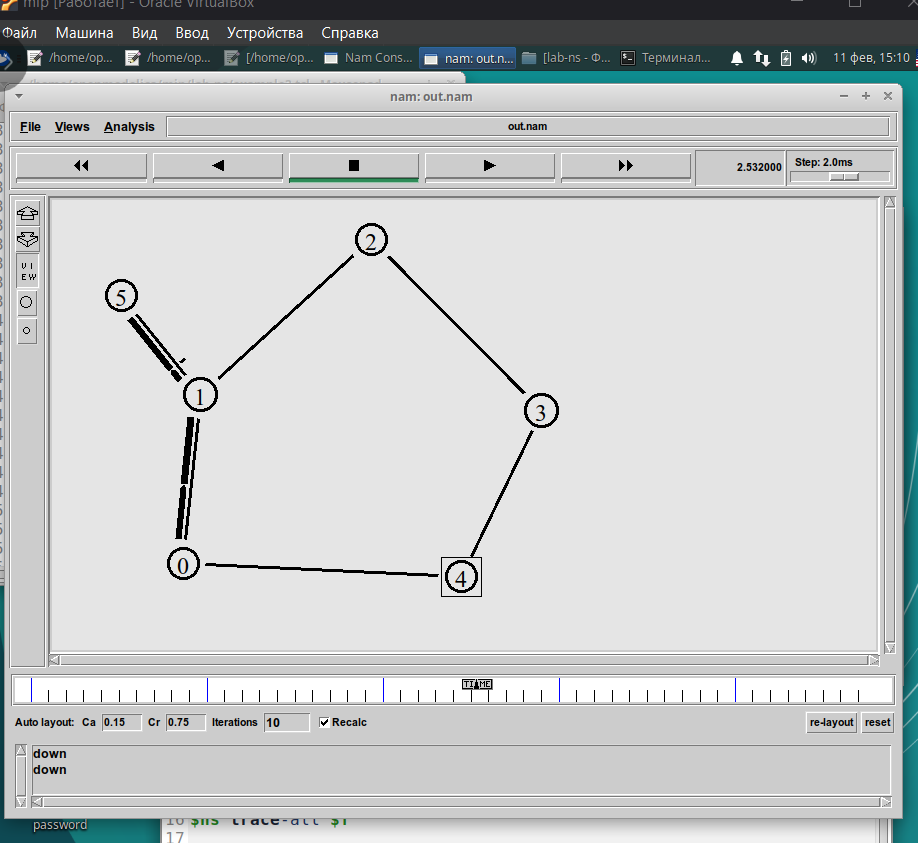


Figure 22: Передача данных после восстановления соединения между 0 и 1 узлами

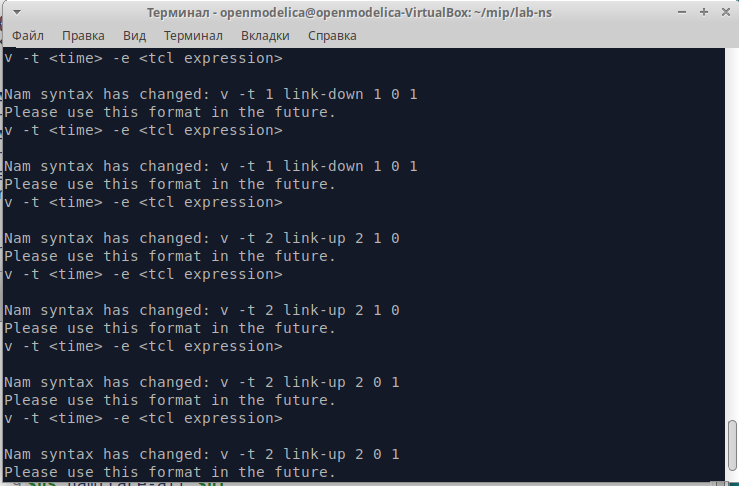


Figure 23: Сообщения в терминале

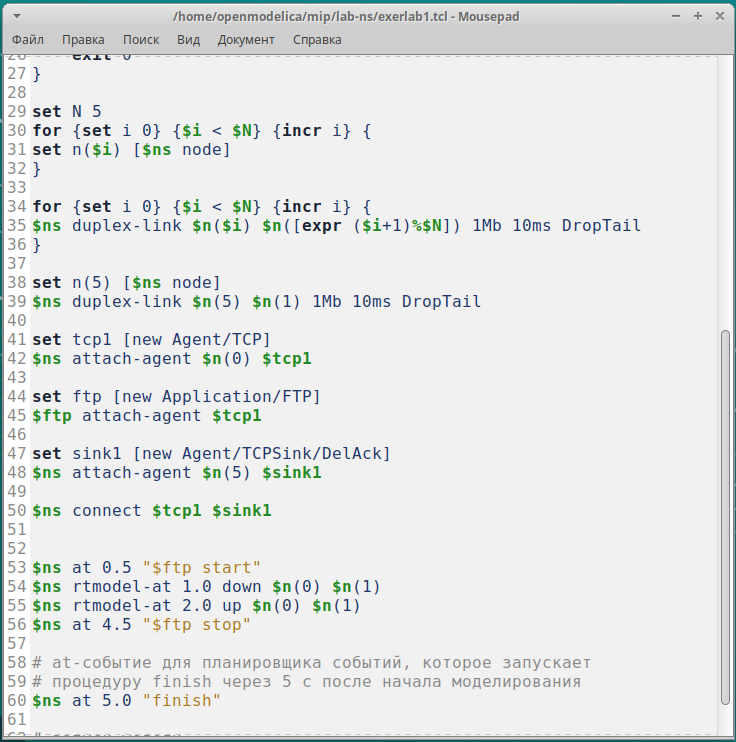


Figure 24: Вид скрипта упражнения

1. Листинг упражнения:

# создание объекта Simulator  
set ns [new Simulator]  
$ns rtproto DV  
  
# открытие на запись файла out.nam для визуализатора nam  
set nf [open out.nam w]  
  
# все результаты моделирования будут записаны в переменную nf  
$ns namtrace-all $nf  
  
# открытие на запись файла трассировки out.tr  
# для регистрации всех событий  
set f [open out.tr w]  
  
# все регистрируемые события будут записаны в переменную f  
$ns trace-all $f  
  
# процедура finish закрывает файлы трассировки  
# и запускает визуализатор nam  
proc finish {} {  
 global ns f nf # объявление глобальных переменных  
 # запуск nam в фоновом режиме  
 close $f  
 close $nf  
 exec nam out.nam &  
 exit 0  
}  
  
set N 5  
for {set i 0} {$i < $N} {incr i} {  
set n($i) [$ns node]  
}  
  
for {set i 0} {$i < $N} {incr i} {  
$ns duplex-link $n($i) $n([expr ($i+1)%$N]) 1Mb 10ms DropTail  
}  
  
set n(5) [$ns node]  
$ns duplex-link $n(5) $n(1) 1Mb 10ms DropTail  
  
set tcp1 [new Agent/TCP]  
$ns attach-agent $n(0) $tcp1  
  
set ftp [new Application/FTP]  
$ftp attach-agent $tcp1  
  
set sink1 [new Agent/TCPSink/DelAck]  
$ns attach-agent $n(5) $sink1  
  
$ns connect $tcp1 $sink1  
  
  
$ns at 0.5 "$ftp start"  
$ns rtmodel-at 1.0 down $n(0) $n(1)  
$ns rtmodel-at 2.0 up $n(0) $n(1)  
$ns at 4.5 "$ftp stop"  
  
# at-событие для планировщика событий, которое запускает  
# процедуру finish через 5 с после начала моделирования  
$ns at 5.0 "finish"  
  
# запуск модели  
$ns run

# 5 Выводы

В ходе данной работы я приобрела практические навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также выполнила анализ полученных результатов моделирования.

# Список литературы

1. [Руководство к лабораторной работе](https://esystem.rudn.ru/mod/resource/view.php?id=1223329)