

Отчет по лабораторной работе №1

Дисциплина: Имитационное моделирование

Лобанова Полина Иннокентьевна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	8
4	Выводы	22

Список иллюстраций

3.1	Создание каталогов и файла	8
3.2	Заполнение шаблона	9
3.3	Запуск симулятора	9
3.4	Аниматора пат	10
3.5	Копирование шаблона	10
3.6	Заполнение файла	11
3.7	Запуск симулятора	11
3.8	Аниматора пат	12
3.9	Копирование шаблона	12
3.10	Заполнение файла	13
3.11	Запуск симулятора	13
3.12	Аниматора пат	14
3.13	Копирование шаблона	14
3.14	Заполнение файла	15
3.15	Передача данных по кратчайшему пути	16
3.16	Прерывание соединения	17
3.17	Передача данных по резервному маршруту	18
3.18	Копирование шаблона	18
3.19	Заполнение файла	19
3.20	Заполнение файла	19
3.21	Запуск симулятора	19
3.22	Передача данных по кратчайшему пути	20
3.23	Прерывание соединения	20
3.24	Передача данных по резервному маршруту	21

Список таблиц

1 Цель работы

Приобретение навыков моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также анализ полученных результатов моделирования.

2 Задание

1. Создать шаблон сценария для NS-2.
2. Требуется смоделировать сеть передачи данных, состоящую из двух узлов, соединённых дуплексной линией связи с полосой пропускания 2 Мб/с и задержкой 10 мс, очередь с обслуживанием типа DropTail. От одного узла к другому по протоколу UDP осуществляется передача пакетов, размером 500 байт, с постоянной скоростью 200 пакетов в секунду.
3. Описание моделируемой сети:
 - сеть состоит из 4 узлов (n0, n1, n2, n3);
 - между узлами n0 и n2, n1 и n2 установлено дуплексное соединение с пропускной способностью 2 Мбит/с и задержкой 10 мс;
 - между узлами n2 и n3 установлено дуплексное соединение с пропускной способностью 1,7 Мбит/с и задержкой 20 мс;
 - каждый узел использует очередь с дисциплиной DropTail для накопления пакетов, максимальный размер которой составляет 10;
 - TCP-источник на узле n0 подключается к TCP-приёмнику на узле n3 (по умолчанию, максимальный размер пакета, который TCP-агент может генерировать, равняется 1KByte)
 - TCP-приёмник генерирует и отправляет ACK пакеты отправителю и откидывает полученные пакеты;
 - UDP-агент, который подсоединён к узлу n1, подключён к null-агенту на узле n3 (null-агент просто откидывает пакеты);

- генераторы трафика ftp и cbr прикреплены к TCP и UDP агентам соответственно;

- генератор cbr генерирует пакеты размером 1 Кбайт со скоростью 1 Мбит/с;
- 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.

4. Требуется построить модель передачи данных по сети с кольцевой топологией и динамической маршрутизацией пакетов:

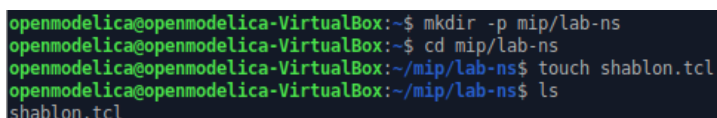
- сеть состоит из 7 узлов, соединённых в кольцо;
- данные передаются от узла $n(0)$ к узлу $n(3)$ по кратчайшему пути;
- с 1 по 2 секунду модельного времени происходит разрыв соединения между узлами $n(1)$ и $n(2)$;
- при разрыве соединения маршрут передачи данных должен измениться на резервный.

5. Внесите следующие изменения в реализацию примера с кольцевой топологией сети:

- 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.

3 Выполнение лабораторной работы

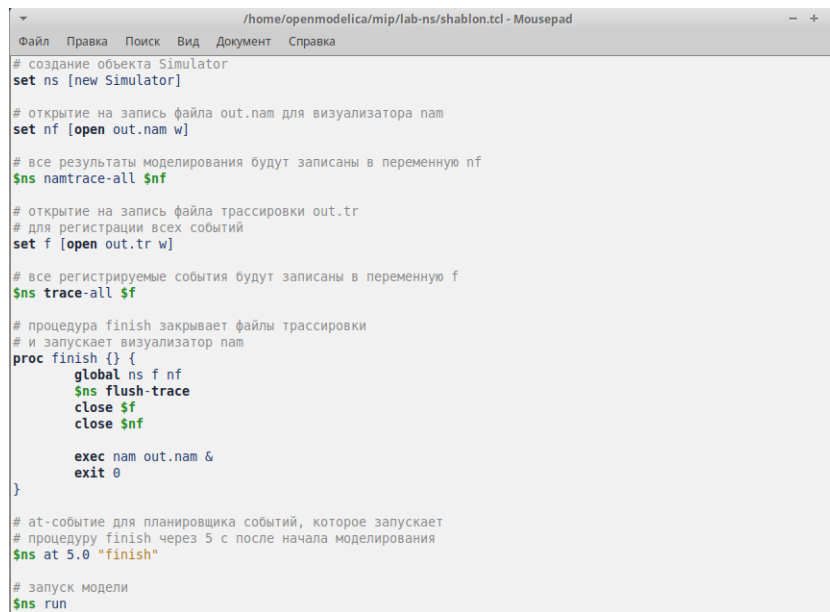
1. В своём рабочем каталоге создала директорию `mip`, к которой будут выполняться лабораторные работы. Внутри `mip` создала директорию `lab-ns`, а в ней файл `shablon.tcl`.



```
openmodelica@openmodelica-VirtualBox:~$ mkdir -p mip/lab-ns
openmodelica@openmodelica-VirtualBox:~$ cd mip/lab-ns
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ touch shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ls
shablon.tcl
```

Рис. 3.1: Создание каталогов и файла

2. Открыла на редактирование файл `shablon.tcl`. Сначала создала объект типа `Simulator`. Затем создала переменную `nf` и указала, что требуется открыть на запись `nam`-файл для регистрации выходных результатов моделирования. Далее создала переменную `f` и открыла на запись файл трассировки для регистрации всех событий модели. После этого добавила процедуру `finish`, которая закрывает файлы трассировки и запускает `nam`. Наконец, с помощью команды `at` указала планировщику событий, что процедуру `finish` следует запустить через 5 с после начала моделирования, после чего запустить симулятор `ns`.



```
/home/openmodelica/mip/lab-ns/shablon.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка

# создание объекта Simulator
set ns [new Simulator]

# 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
    $ns flush-trace
    close $f
    close $nf

    exec nam out.nam &
    exit 0
}

# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"

# запуск модели
$ns run
```

Рис. 3.2: Заполнение шаблона

3. Сохранив изменения в отредактированном файле shablon.tcl и закрыв его, запустила симулятор.



```
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ns shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$
```

Рис. 3.3: Запуск симулятора

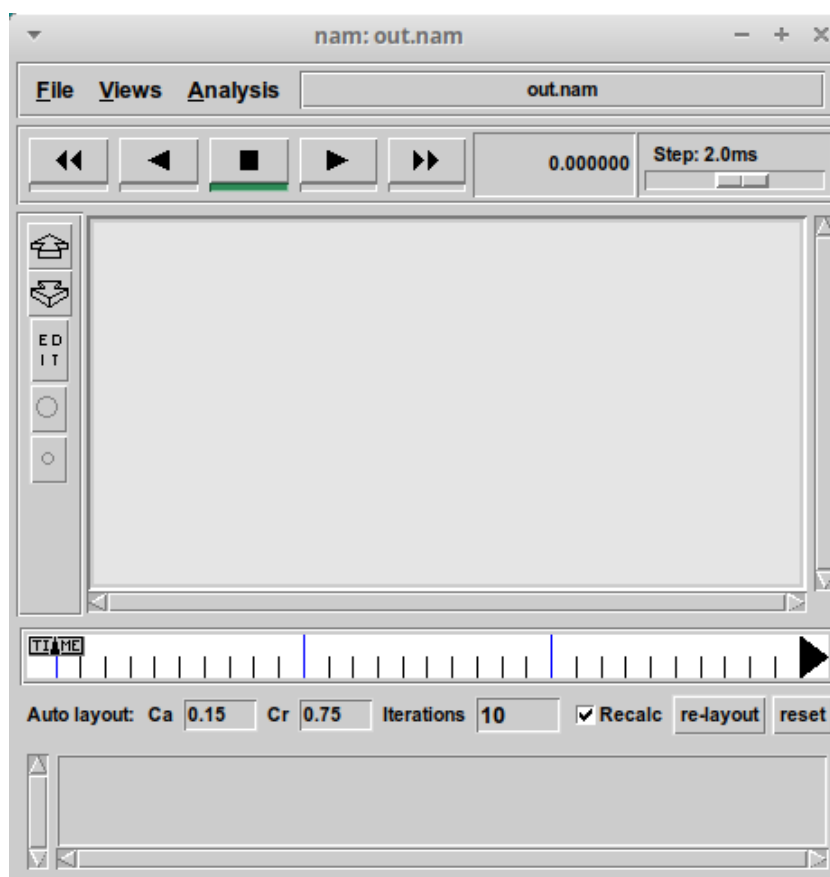


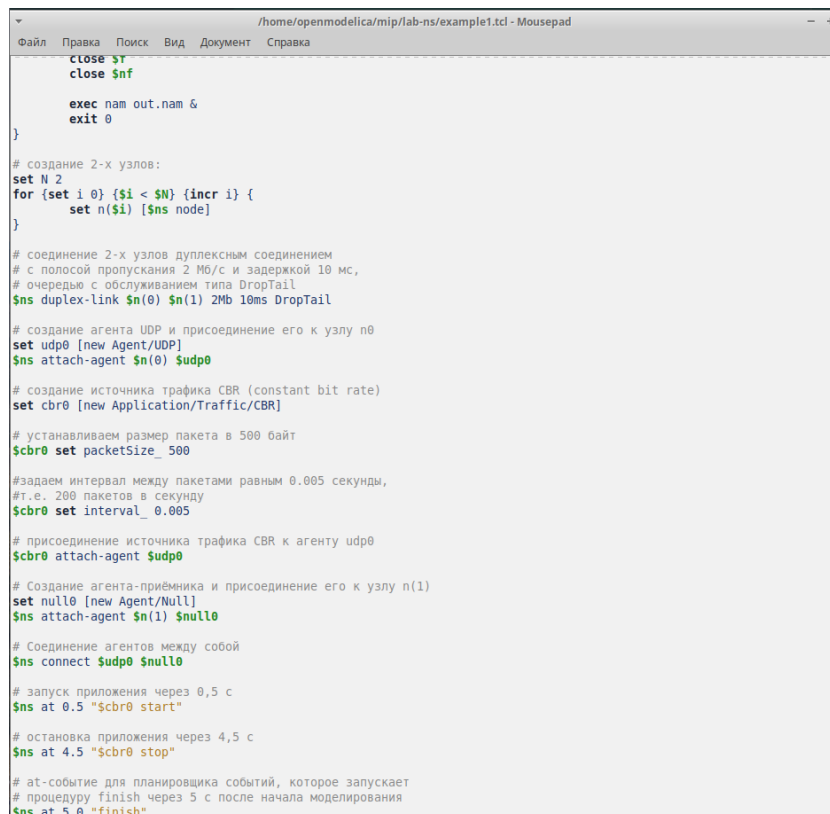
Рис. 3.4: Аниматора *nam*

4. Скопировала содержимое созданного шаблона в новый файл.

```
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ cp shablon.tcl example1.tcl
```

Рис. 3.5: Копирование шаблона

5. Добавила в него до строки \$ns at 5.0 “finish” описание топологии сети. Создала агенты для генерации и приёма трафика. Далее создала Null-агент, который работает как приёмник трафика, и прикрепила его к узлу n1. Соединила агенты между собой. Для запуска и остановки приложения CBR добавляются at-события в планировщик событий.



```
close $f
close $nf

exec nam out.nam &
exit 0
}

# создание 2-х узлов:
set N 2
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

# соединение 2-х узлов дуплексным соединением
# с полосой пропускания 2 Мб/с и задержкой 10 мс,
# очередь с обслуживанием типа DropTail
$ns duplex-link $n(0) $n(1) 2Mb 10ms DropTail

# создание агента UDP и присоединение его к узлу n0
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0

# создание источника трафика CBR (constant bit rate)
set cbr0 [new Application/Traffic/CBR]

# устанавливаем размер пакета в 500 байт
$cbr0 set packetSize_ 500

#задаем интервал между пакетами равным 0.005 секунды,
#т.е. 200 пакетов в секунду
$cbr0 set interval_ 0.005

# присоединение источника трафика CBR к агенту udp0
$cbr0 attach-agent $udp0

# Создание агента-приёмника и присоединение его к узлу n(1)
set null0 [new Agent/Null]
$ns attach-agent $n(1) $null0

# Соединение агентов между собой
$ns connect $udp0 $null0

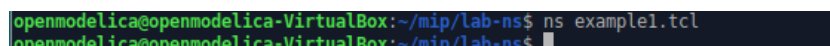
# запуск приложения через 0,5 с
$ns at 0.5 "$cbr0 start"

# остановка приложения через 4,5 с
$ns at 4.5 "$cbr0 stop"

# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"
```

Рис. 3.6: Заполнение файла

6. Сохранила изменения в отредактированном файле и запустила симулятор.



```
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ns example1.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$
```

Рис. 3.7: Запуск симулятора

7. Получила в качестве результата запуск аниматора nam в фоновом режиме.

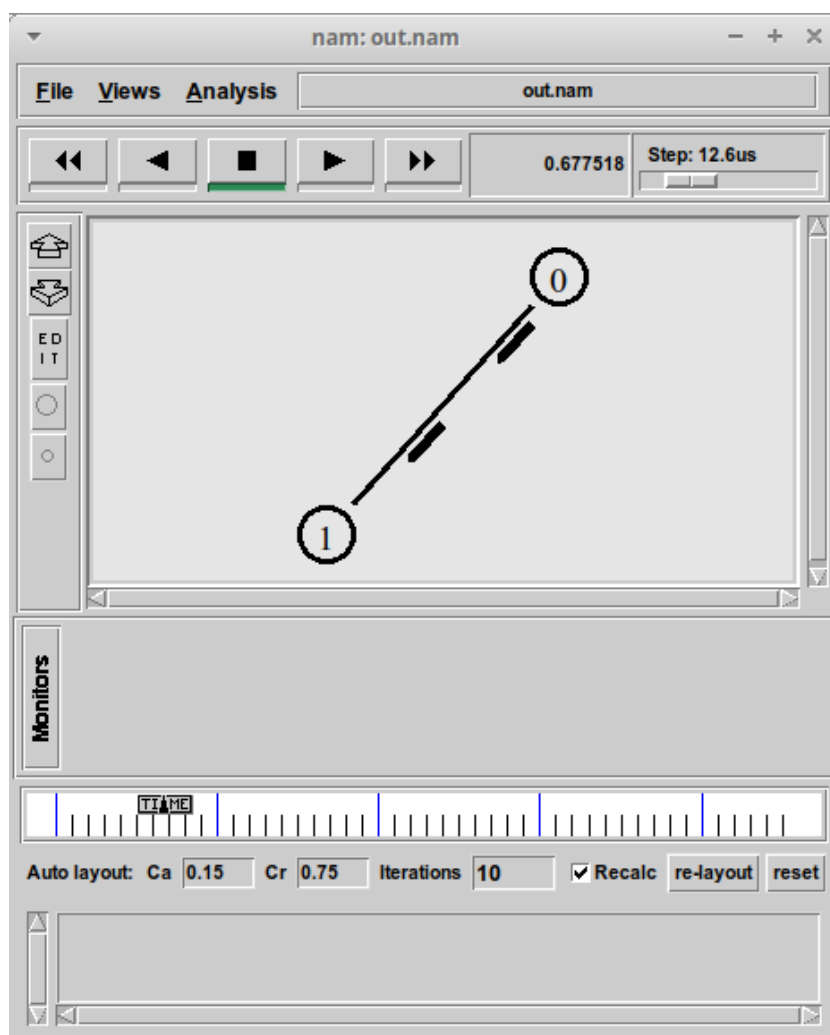


Рис. 3.8: Аниматора *nam*

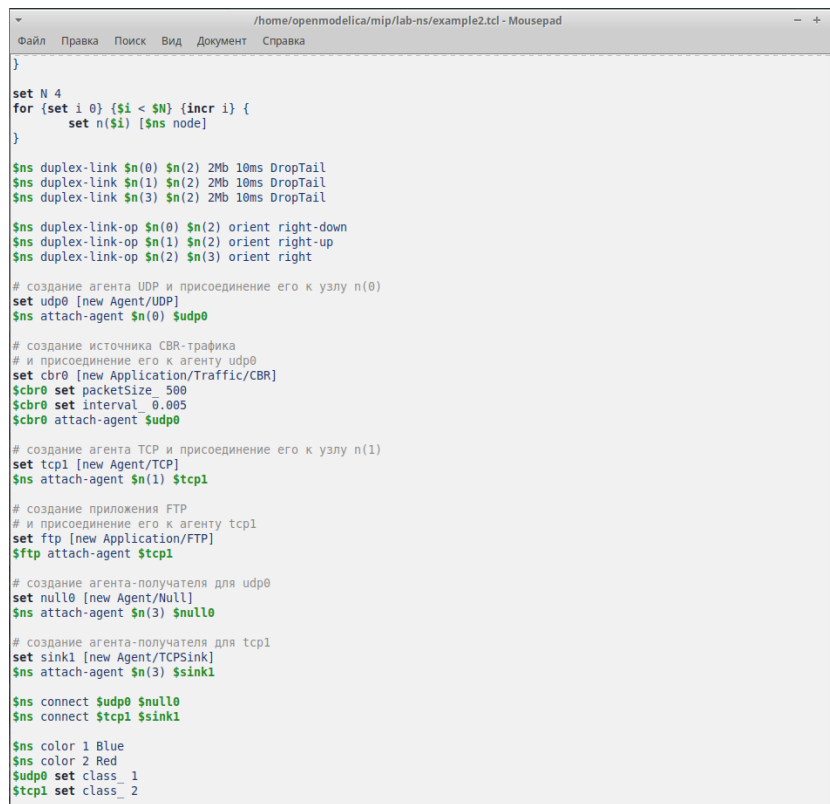
8. Скопировала содержимое созданного шаблона в новый файл.

```
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ cp shablon.tcl example2.tcl
```

Рис. 3.9: Копирование шаблона

9. Создала 4 узла и 3 дуплексных соединения с указанием направления. Создала агент UDP с прикреплённым к нему источником CBR и агент TCP с прикреплённым к нему приложением FTP. Создала агенты-получатели. Соединила агенты `udr0` и `tcp1` и их получателей. Задала описание цвета

каждого потока. Добавила отслеживание событий в очереди, наложение ограничения на размер очереди и at-события.



```
}/home/openmodelica/mip/lab-ns/example2.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка

}

set N 4
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

$ns duplex-link $n(0) $n(2) 2Mb 10ms DropTail
$ns duplex-link $n(1) $n(2) 2Mb 10ms DropTail
$ns duplex-link $n(3) $n(2) 2Mb 10ms DropTail

$ns duplex-link-op $n(0) $n(2) orient right-down
$ns duplex-link-op $n(1) $n(2) orient right-up
$ns duplex-link-op $n(2) $n(3) orient right

# создание агента UDP и присоединение его к узлу n(0)
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0

# создание источника CBR-трафика
# и присоединение его к агенту udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize 500
$cbr0 set interval 0.005
$cbr0 attach-agent $udp0

# создание агента TCP и присоединение его к узлу n(1)
set tcp1 [new Agent/TCP]
$ns attach-agent $n(1) $tcp1

# создание приложения FTP
# и присоединение его к агенту tcp1
set ftp [new Application/FTP]
$ftp attach-agent $tcp1

# создание агента-получателя для udp0
set null0 [new Agent/Null]
$ns attach-agent $n(3) $null0

# создание агента-получателя для tcp1
set sink1 [new Agent/TCPSink]
$ns attach-agent $n(3) $sink1

$ns connect $udp0 $null0
$ns connect $tcp1 $sink1

$ns color 1 Blue
$ns color 2 Red
$udp0 set class_1
$tcp1 set class_2
```

Рис. 3.10: Заполнение файла

10. Сохранив изменения в отредактированном файле и запустив симулятор, получила анимированный результат моделирования.



```
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ns example2.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$
```

Рис. 3.11: Запуск симулятора

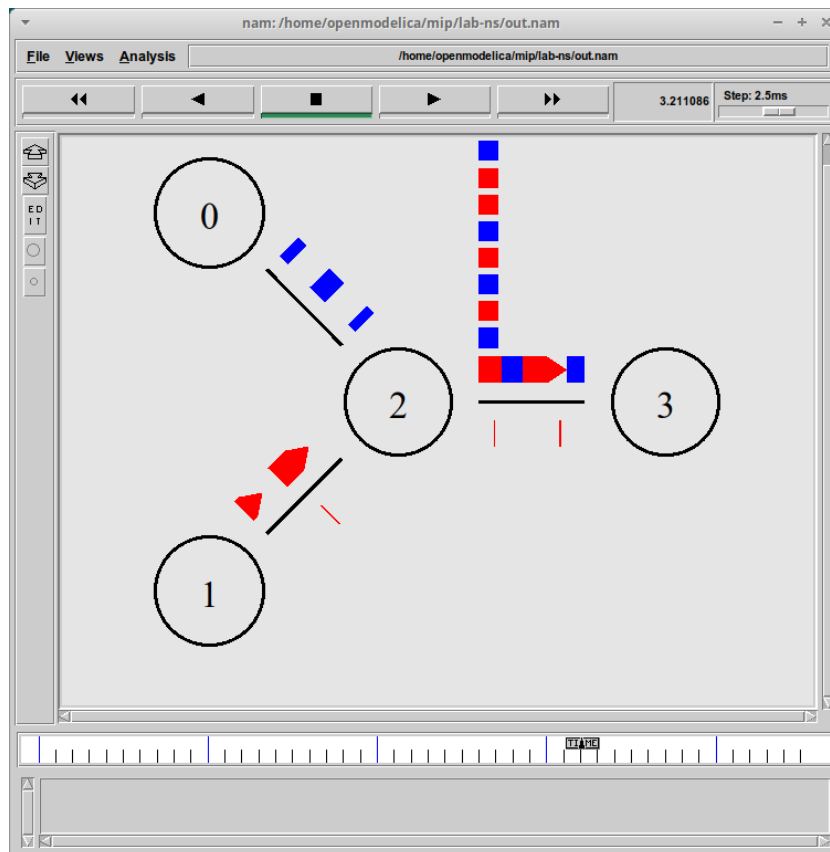


Рис. 3.12: Аниматора *nam*

11. Скопировала содержимое созданного шаблона в новый файл.

```
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ cp shablon.tcl example3.tcl
```

Рис. 3.13: Копирование шаблона

12. Описала топологию моделируемой сети. Далее соединила узлы так, чтобы создать круговую топологию. Задала передачу данных от узла $n(0)$ к узлу $n(3)$. Добавила команду разрыва соединения между узлами $n(1)$ и $n(2)$ на время в одну секунду, а также время начала и окончания передачи данных. Добавила в начало скрипта после команды создания объекта Simulator: `$ns rtproto DV`.

```

# 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
    $ns flush-trace
    close $f
    close $nf

    exec nam out.nam &
    exit 0
}

set N 7
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 udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set cbr0 [new Agent/CBR]
$ns attach-agent $n(0) $cbr0
$cbr0 set packetSize 500
$cbr0 set interval 0.005
set null0 [new Agent/Null]
$ns attach-agent $n(3) $null0
$ns connect $cbr0 $null0

$ns at 0.5 "$cbr0 start"
$ns rtmodel-at 1.0 down $n(1) $n(2)
$ns rtmodel-at 2.0 up $n(1) $n(2)
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"

# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования

```

Рис. 3.14: Заполнение файла

13. Сохранив изменения в отредактированном файле и запустив симулятор, получила анимированный результат моделирования.

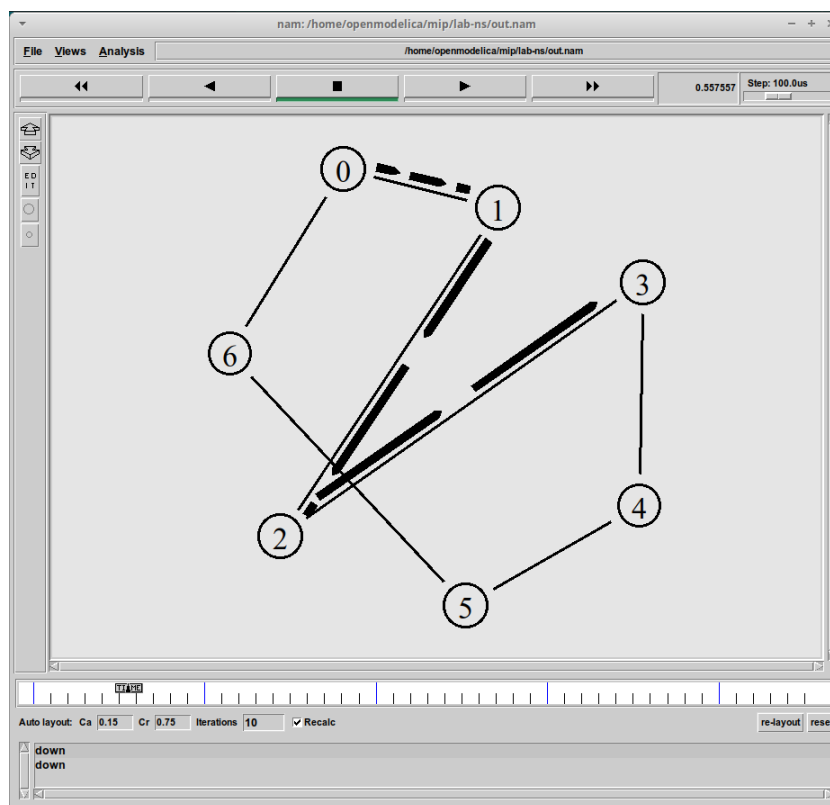


Рис. 3.15: Передача данных по кратчайшему пути

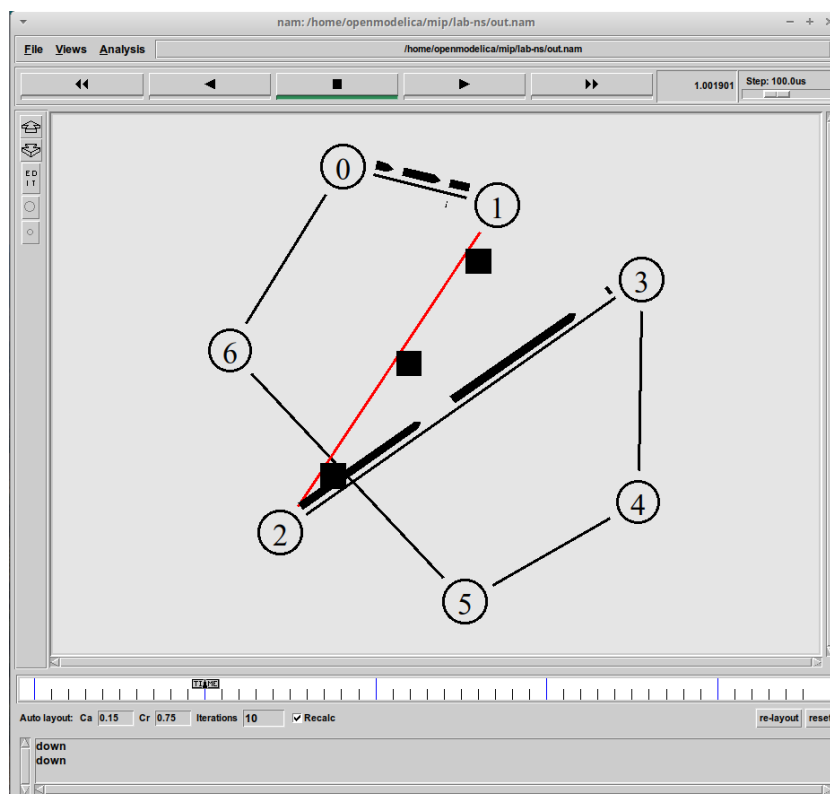


Рис. 3.16: Прерывание соединения

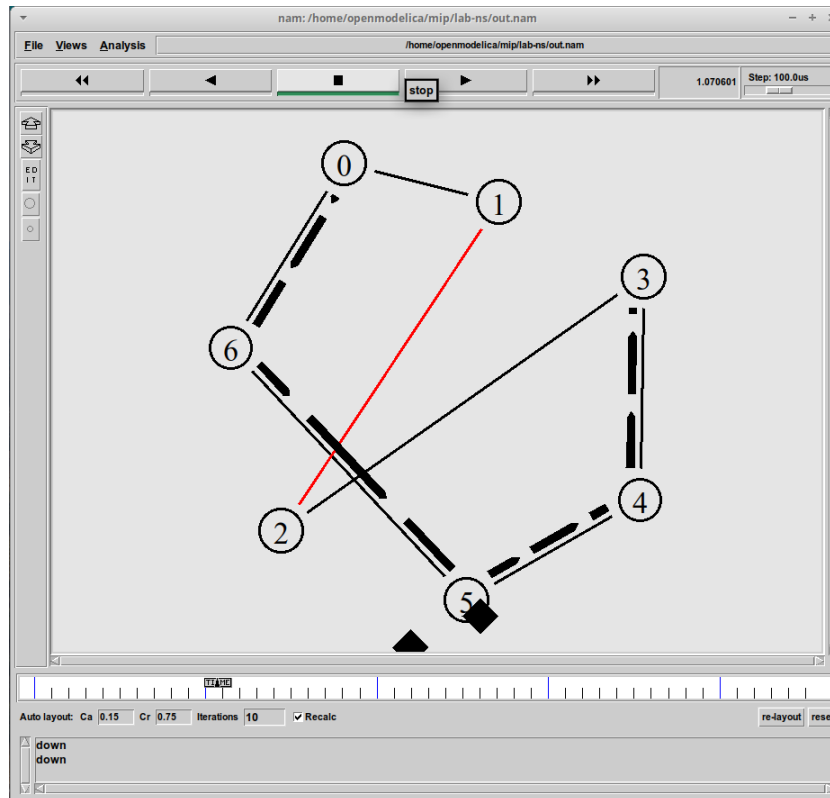


Рис. 3.17: Передача данных по резервному маршруту

14. Скопировала содержимое созданного шаблона в новый файл.

```
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ cp shablon.tcl ex1.tcl
```

Рис. 3.18: Копирование шаблона

15. Создала 5 узлов и соединила их так, чтобы создать круговую топологию. Создала еще один узел(n(5)) и соединила его с узлом n(1). Задала передачу данных от узла n(0) к узлу n(5). Создала агент TCP (тип Newreno) с прикрепленным к нему приложением FTP. Создала агент-получатель (TCPSink-объект типа DelAck). Соединила агент tcp1 и его получателя. Добавила команду разрыва соединения между узлами n(0) и n(1) на время в одну секунду, а также время начала и окончания передачи данных. Добавила в начало скрипта после команды создания объекта Simulator: \$ns rtproto DV.

```

/home/openmodelica/mip/lab-ns/ex1.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка

# 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
    $ns flush-trace
    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(1) $n(5) 2Mb 10ms DropTail

set tcp1 [new Agent/TCP/Newreno]
$ns attach-agent $n(0) $tcp1

set sink1 [new Agent/TCP/Sink/DelAck]
$ns attach-agent $n(5) $sink1

set ftp [new Application/FTP]
$ftp attach-agent $tcp1

```

Рис. 3.19: Заполнение файла

```

$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

```

Рис. 3.20: Заполнение файла

16. Сохранив изменения в отредактированном файле и запустив симулятор, получила анимированный результат моделирования.

```

openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ns ex1.tcl

```

Рис. 3.21: Запуск симулятора

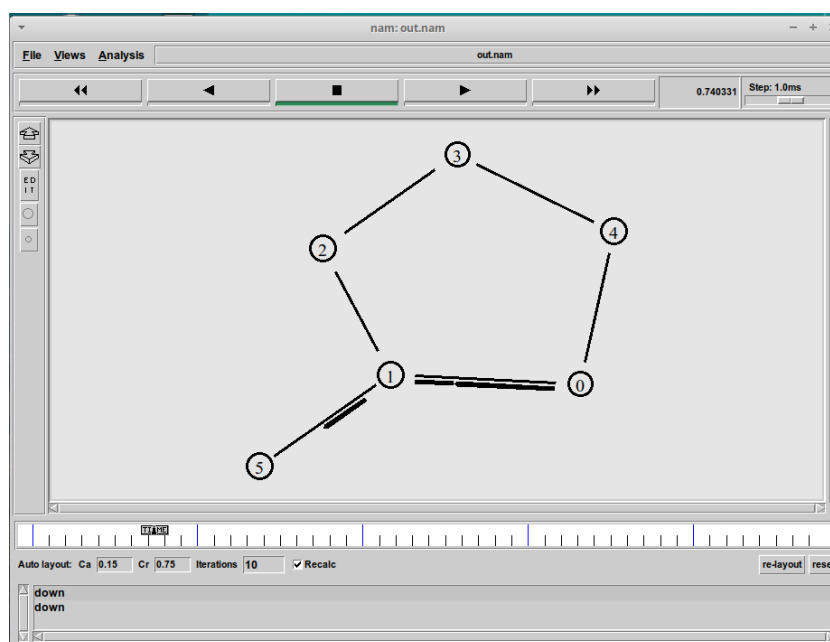


Рис. 3.22: Передача данных по кратчайшему пути

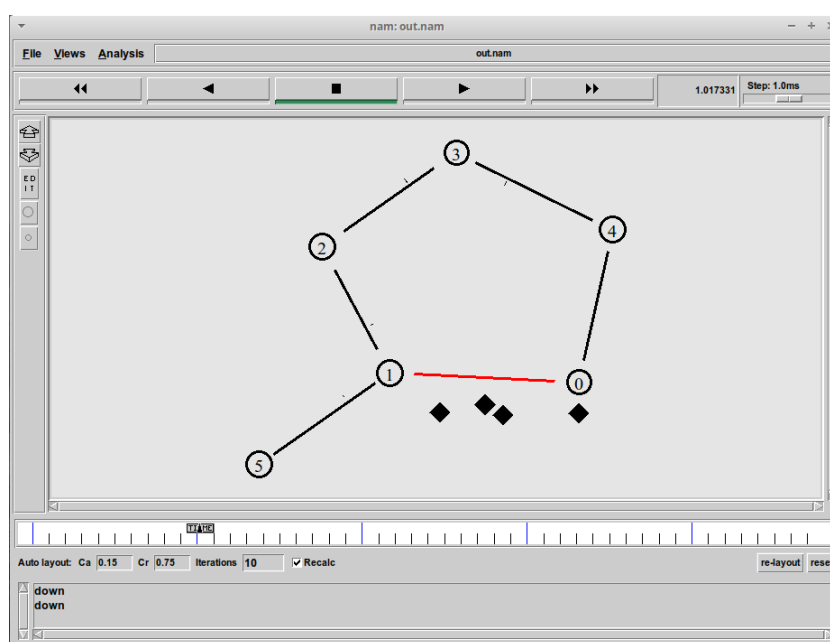


Рис. 3.23: Прерывание соединения

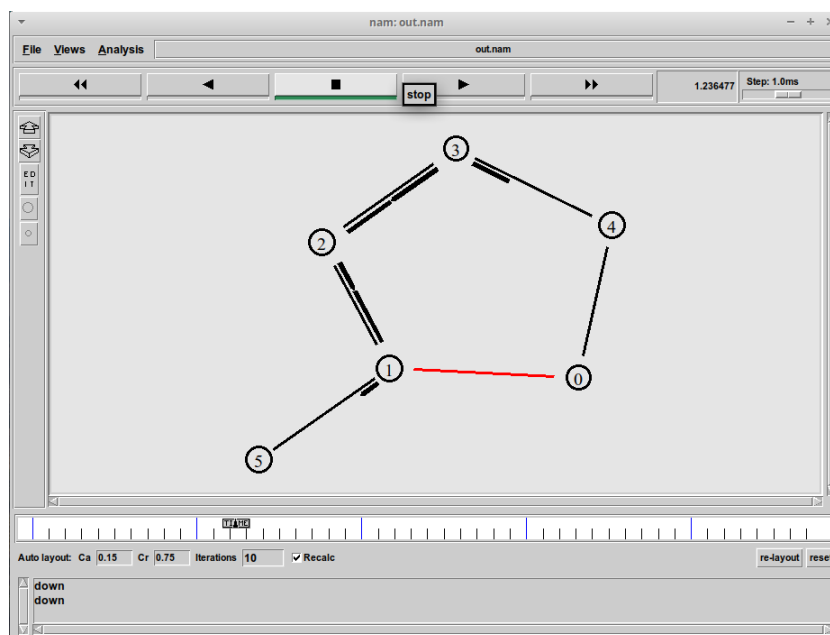


Рис. 3.24: Передача данных по резервному маршруту

4 Выводы

Я приобрела навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также анализ полученных результатов моделирования.