|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_\_Информатика, искусственный интеллект и системы управления\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_\_\_Программное обеспечение ЭВМ и информационные технологии\_\_\_\_\_\_\_\_\_

**ОТЧЕТ**

**по Лабораторной работе №1**

**по курсу**

**«Математические основы верификации ПО»**

**Тема**

**«Знакомство с языком Promela»**

Студент \_\_\_ИУ7-41М\_\_\_\_ \_\_\_\_\_Карпухин А.С.\_\_\_\_

(Группа) (И.О.Фамилия)

Преподаватель \_\_\_\_Кузнецова О.В.\_\_\_\_

(И.О.Фамилия)

*2022 г.*

**Задание**

Для небольшого фрагмента программы необходимо описать модель на Promela и изучить её.

**Описание модели**

В качестве примера была описана модель простого вымышленного протокола передачи данных с механизмом установки соединения, подобным TCP. Передача осуществляется между двумя сторонами — отправителем и получателем. Отправитель инициирует установку соединения, отправляет заданное число сообщений с данными и затем выполняет закрытие соединения.

Графы состояний отправителя и получателя приведены на рисунке 1.

Участники обмена данными могут находится в следующих состояниях:

1 — начальное состояние, соединение не установлено

2 — отправитель отправил EST; получатель получил EST

3 — отправитель получил ACK/EST; получатель отпрвил ACK/EST

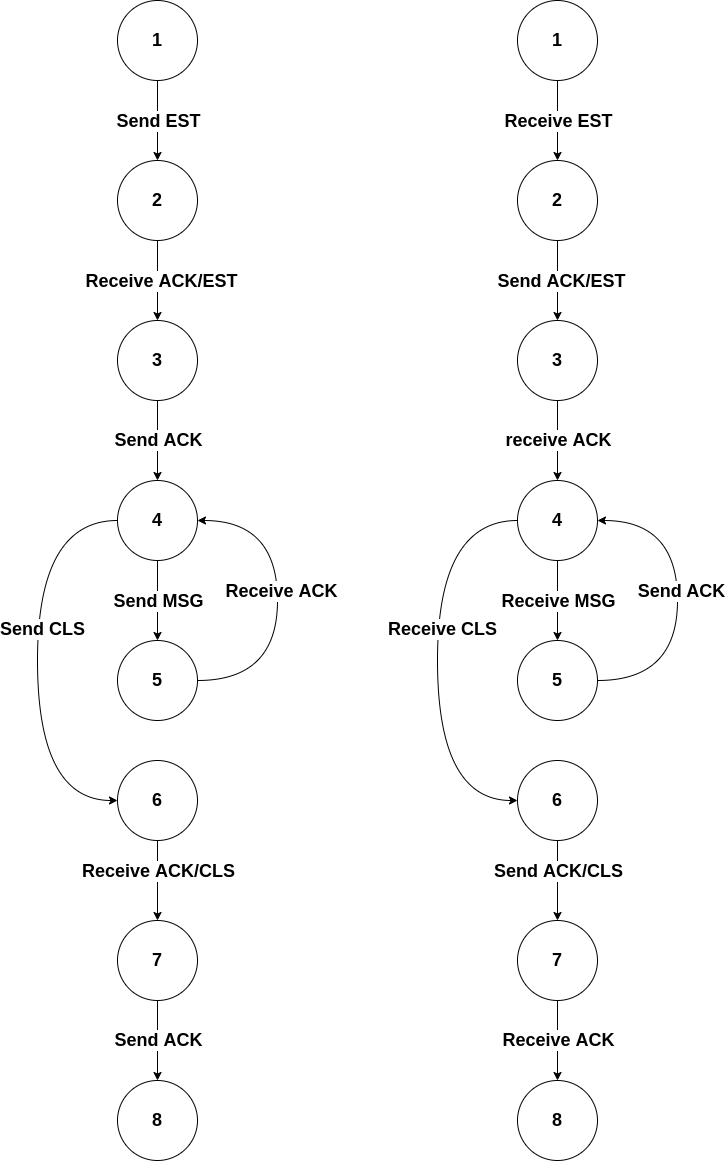
4 — отправитель отправил ACK; получатель получил ACK; соединение установлено

5 — отправитель послал MSG; получатель получил MSG

6 — закрытие соединения — отправитель послал CLS; получатель принял CLS

7 — отправитель получил ACK/CLS; получатель послал ACK/CLS

8 — отправитель послал ACK; получатель принял ACK; соединение закрыто

Рисунок 1 - граф состояний отправителя (слева) и получателя (справа)

**Листинги кода**

В листингах 1 и 2 приведено описание моделей отправителя и получателя соответственно.

Листинг 1 — Код модели отправителя.

|  |
| --- |
| proctype Sender(chan ch; int msgCount)  {  byte sendbit, recvbit;  printf("SENDER: running, pid=%d\n", \_pid);  rand(sendbit);  printf("SENDER: sending EST\n");  ch ! EST, sendbit;  printf("SENDER: waiting for ACK\n");  ch ? ACK, recvbit;  if  :: recvbit == sendbit ->  printf("SENDER: ACK received, waiting for EST\n");  ch ? EST, recvbit;  printf("SENDER: EST received, sending ACK\n");  ch ! ACK, recvbit;  printf("SENDER: connection established\n");  do  :: msgCount > 0 ->  rand(sendbit);  printf("SENDER: sending MSG %d\n", sendbit);  ch ! MSG, sendbit;  printf("SENDER: waiting for ACK\n");  ch ? ACK, recvbit;  msgCount = msgCount - 1;  if  :: recvbit == sendbit -> printf("SENDER: ACK received, verification number is valid\n");  :: else -> printf("SENDER: ACK received, but verification number is invalid\n");  fi  :: msgCount == 0 ->  rand(sendbit);  printf("SENDER: end of transmition, sending CLS\n");  ch ! CLS, sendbit;  printf("SENDER: waiting for ACK\n");  ch ? ACK, recvbit;  if  :: recvbit == sendbit ->  printf("SENDER:: ACK received, waiting for CLS\n");  ch ? CLS, recvbit;  ch ! ACK, recvbit;  printf("SENDER: CLS received, clean close\n");  :: else -> printf("SENDER: ACK received, invalid verification number, dirty close\n");  fi  break;  od  :: else -> printf("SENDER: ACK received, invalid verification number, connection was not established\n");  fi  } |

Листинг 2 — Код модели получателя.

|  |
| --- |
| proctype Receiver(chan ch)  {  byte sendbyte, recvbyte;  printf("RECEIVER: running, pid=%d\n", \_pid);  printf("RECEIVER: waiting for EST\n");  ch ? EST, recvbyte;  printf("RECEIVER: EST received, sending ACK\n");  ch ! ACK, recvbyte;  rand(sendbyte);  printf("RECEIVER: sending EST\n");  ch ! EST, sendbyte;  printf("RECEIVER: waiting for ACK\n");  ch ? ACK, recvbyte;  if  :: recvbyte == sendbyte ->  printf("RECEIVER: connection established, waiting for incoming MSG\n");  do  :: ch ? MSG, recvbyte ->  printf("RECEIVER: MSG received with code %d\n", recvbyte);  ch ! ACK, recvbyte;  :: ch ? CLS, recvbyte ->  printf("RECEIVER: CLS received, closing connection ...\n");  ch ! ACK, recvbyte;  rand(sendbyte);  ch ! CLS, sendbyte;  ch ? ACK, recvbyte;  if  :: recvbyte == sendbyte ->  break;  :: else  fi  od  :: else  fi  } |