Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

Верификация и анализ программ

Отчет по лабораторной работе №3

Верификация модели с помощью NuSMV

Выполнил студент гр. 23541/3 Смирнов С.В.

(подпись)

Преподаватель Ицыксон В.М.

(подпись)

“ ” 2019 г.

Санкт – Петербург

2019

## Цель работы

Необходимо построить автоматную модель для следующей задачи:

Парк развлечений, в котором есть три кассы для покупки билетов на аттракционы и три очереди к аттракционам. Для того, чтобы прокатиться на любом аттракционе необходимо отстоять очередь в кассу, приобрести билет и затем отстоять очередь к аттракциону. В случае если касса свободна, человек из очереди подходит к кассе, оплачивает билет, после оплаты клиенты встают в очередь к интересующему аттракциону.

В том случае, когда очередь к аттракционам заполнена, клиент уходит домой.

При описании модели стремиться сделать ее масштабируемой по отношению к клиентам и кассам. За стандартную модель были приняты следующие значения: длина очереди - 3, количество касс – 3.

Проверить следующие утверждения:

* Все кассы, в конечном итоге, освободятся.
* Все очереди, в конечном итоге, освободятся.
* Все клиенты прокатятся на аттракционе или покинут парк
* Если клиент был в очереди, то, в конечном итоге, он прокатится.

## Построение модели

Для реализации описанной системы был построен конечный автомат движения клиентов (рис.2) и конечный автомат очередей в кассу и на аттракцион(рис.1).

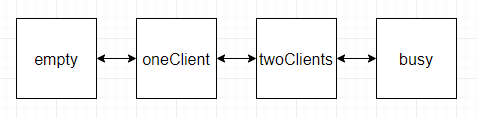


Рис.1. Конечный автомат очереди (для очереди в кассу и ожидания аттракциона)

Состояния клиента:

* WAIT\_PAY – ожидание в очереди кассы
* PAY - оплата
* WAIT\_ACTION – ожидание в очереди аттракциона
* ACTION – катание на аттракционе- позитивное завершение
* GO\_AWAY - негативное завершение- у клиента кончилось терпение

Клиент встает в очередь.

WAIT\_PAY

ACTION

10

PAY

GO\_AWAY

WAIT

ACTION

Рис.2. Конечный автомат движения клиента

Ниже представлена реализация описанной модели на языке PROMELA:

MODULE main

VAR

client1 : process clients (0, clientsStates, pay\_queue, wait\_queue);

client2 : process clients (1, clientsStates, pay\_queue, wait\_queue);

client3 : process clients (2, clientsStates, pay\_queue, wait\_queue);

client4 : process clients (3, clientsStates, pay\_queue, wait\_queue);

clientsStates : array 0..4 of {waitPay, pay, waitAction, action, goAway};

wait\_queue : {empty, oneClient, twoClients, busy};

pay\_queue : {empty, oneClient, twoClients, busy};

ASSIGN

init(wait\_queue) := empty;

init(pay\_queue) := empty;

MODULE clients(id, clientsStates, pay\_queue, wait\_queue)

DEFINE

state := clientsStates[id];

ASSIGN

init (clientsStates[id]) := waitPay;

next(pay\_queue) :=

case

state = waitPay & pay\_queue = empty : oneClient;

state = waitPay & pay\_queue = oneClient : twoClients;

state = waitPay & pay\_queue = twoClients : busy;

state = waitAction & pay\_queue = busy : twoClients;

state = waitAction & pay\_queue = twoClients : oneClient;

state = waitAction & pay\_queue = oneClient : empty;

TRUE : pay\_queue;

esac;

next(wait\_queue) :=

case

state = action & wait\_queue = busy : twoClients;

state = action & wait\_queue = twoClients : oneClient;

state = action & wait\_queue = oneClient : empty;

state = waitAction & wait\_queue = twoClients : busy ;

state = waitAction & wait\_queue = oneClient : twoClients;

state = waitAction & wait\_queue = empty : oneClient;

TRUE : wait\_queue;

esac;

next (clientsStates[id]) :=

case

state = waitPay & pay\_queue != busy : pay;

state = pay & wait\_queue != busy : waitAction;

state = waitAction & wait\_queue = busy : goAway;

state = waitAction & wait\_queue != busy : action;

state = action : goAway;

TRUE : state;

esac;

FAIRNESS

running

LTLSPEC G ((clientsStates[0] = action & (clientsStates[1] = action) & (clientsStates[2] =

action) & (clientsStates[3] != action)) -> wait\_queue = busy)

LTLSPEC G ((clientsStates[0] = pay & (clientsStates[1] = pay) & (clientsStates[2] =

pay) & (clientsStates[3] != pay)) -> pay\_queue = busy)

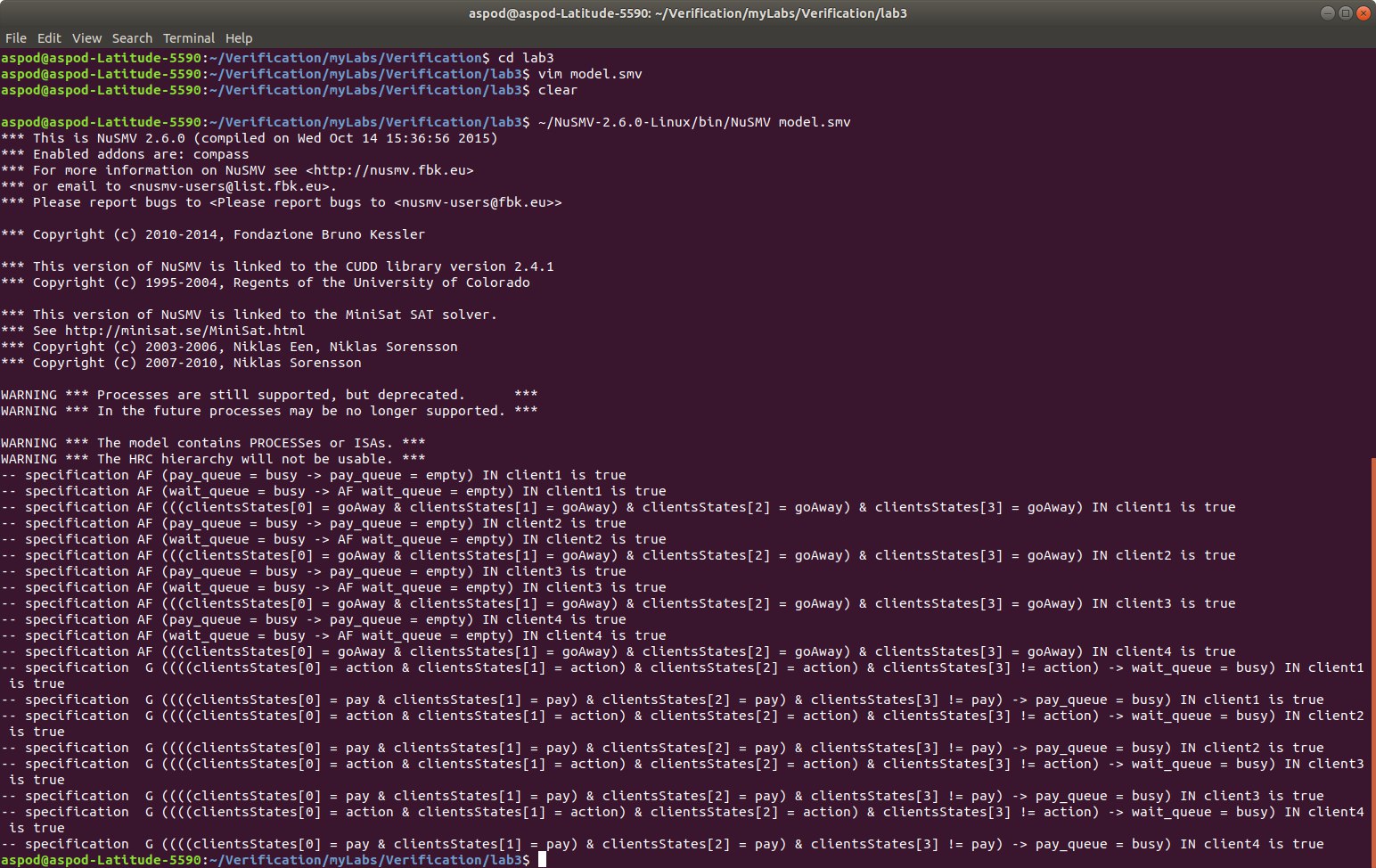
CTLSPEC AF (pay\_queue = busy -> pay\_queue = empty)

CTLSPEC AF (wait\_queue = busy -> AF wait\_queue = empty)

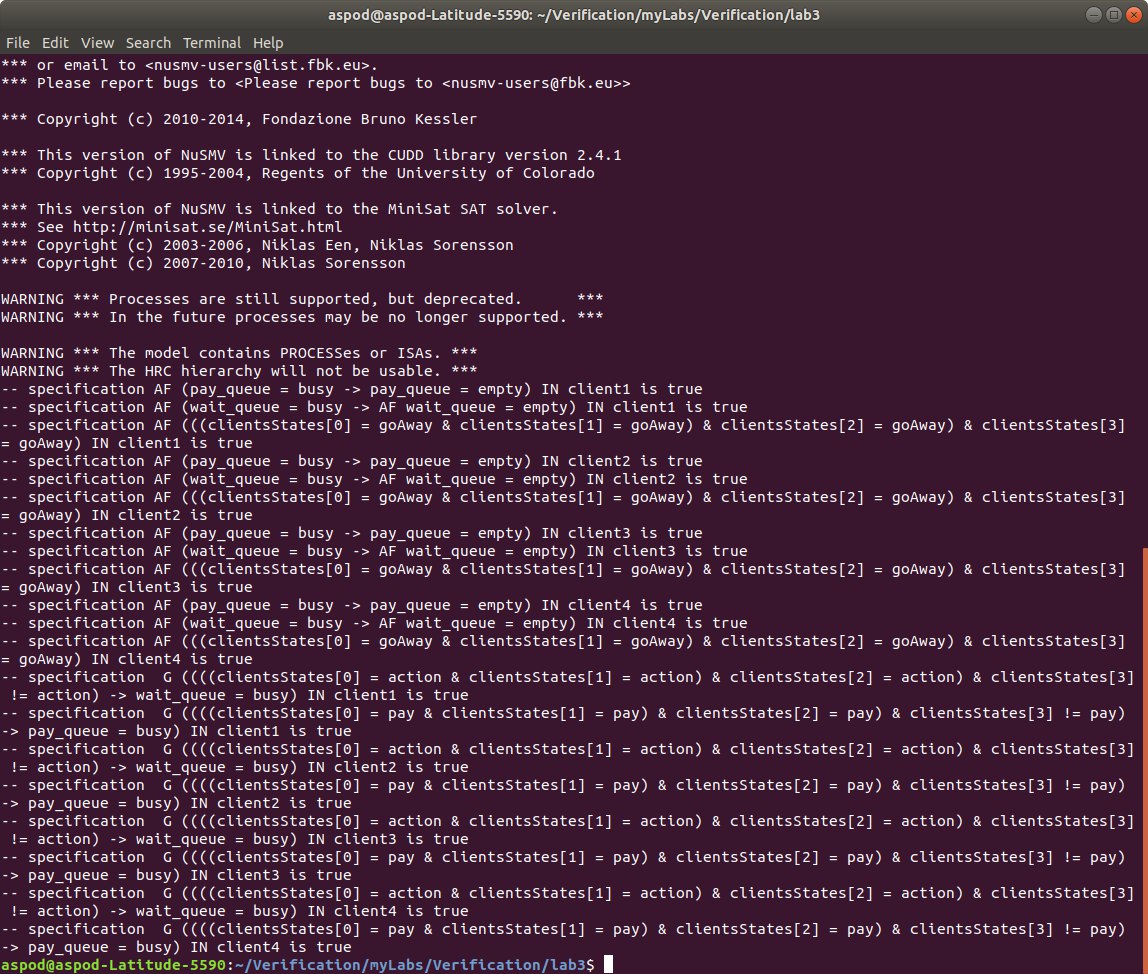
CTLSPEC AF ((clientsStates[0] = goAway) & (clientsStates[1] = goAway) &

(clientsStates[2] = goAway) & (clientsStates[3] = goAway))

Результат запуска:



Результат проверки составленных требований:



## Выводы

В ходе работы была описана и верифицирована модель системы работы парка аттракционов. Для описания спецификаций использовались и LTL и CTL формулы. Верификация была проведена с помощью утилиты NuSMV.

Исходя из опыта работы с утилитой NuSMV, полученного в ходе выполнения данной работы, можно сделать вывод о том, что верификация требует не мало технических ресурсов и занимает значительное количество времени, в том числе на точное создание модели. Однако, она позволяет решить множество задач, которые оказываются трудоемкими для решения человеком.