# SYSC 5104: Modeling Discrete-Event Systems Using DEVS (Fall 2016)

## Assignment 1: Food chain Drive Thru Simulator


**Submitted To: Dr. Gabriel A Wainer** **Submitted By: Harkishan Sohanpal (101026577)**

## Part 1-Conceptual model

Fast Food chains have become an indispensible part of our lives. Most of the food chains try to make their service of providing food quick and efficient. In order to achieve this, the concept of drive thru was invented which became popular for its fast and efficient service. In this Assignment we are going to use DEVS models to test and try a drive thru for fast food restaurant and predict whether the drive thru method of service is going to be efficient for the restaurant.

For the model we are taking there are 10 menu items in the restaurant divided into four categories. Customer will choose from any of the menu items that will be categorized into sections in which they would work in the kitchen. We have divided the kitchen into four sections i.e. Salads, Fryers, Grill and Pans. As soon as the customer comes up to the window and place the order the system takes the order and puts into one of the four categories, there may be combo order but only the ones provided in the menu. Customized combo orders are not allowed by customer in this case. When the order is received the respective sections of the Kitchen are notified about the order and the process of cooking starts. When the order is finished cooking, it goes to the delivery section where it's packed and delivered to the customer. The incoming customer distribution is taken as random normal distribution. The mean and standard deviation will be chosen for different cases.

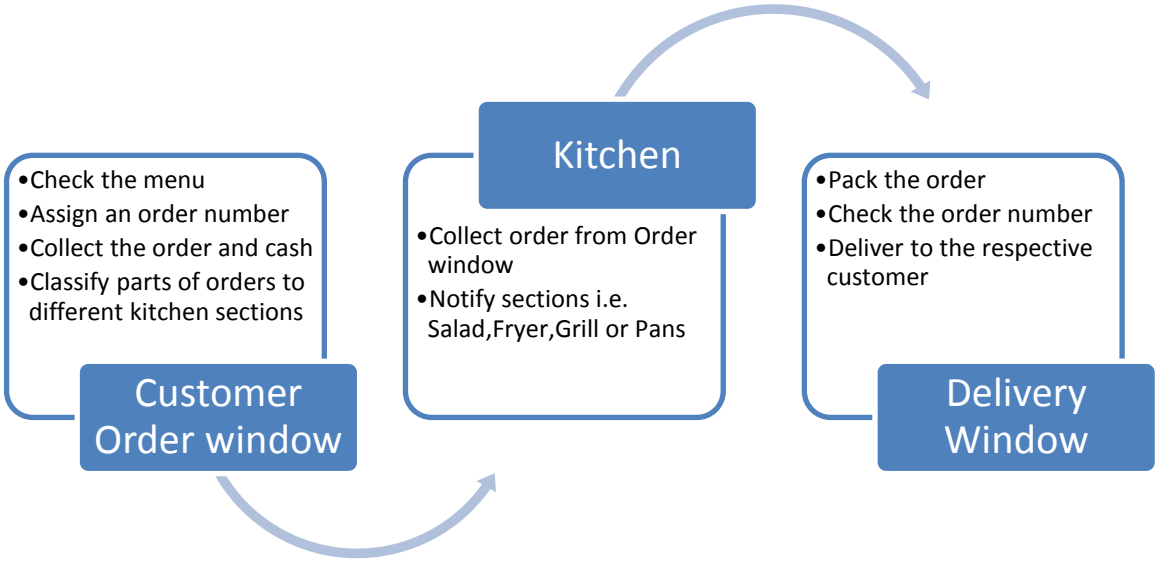



**Fig 1.Basic flow Diagram of the system**

The above figure shows the method by which the system is going to work. There are subdivisions to the sections mentioned in the above figure which will lead to the discrete event model. The atomic and couple models will be structured from there further on.

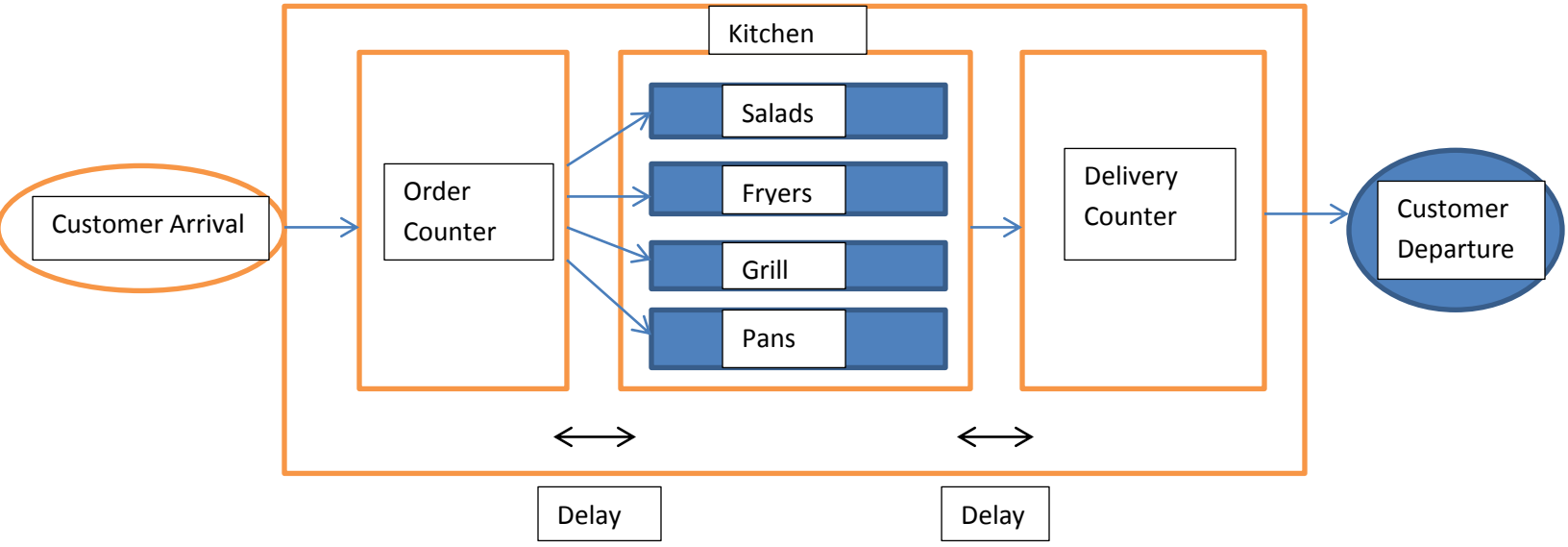



**Fig 2 Discrete event model**

The Discrete event model consists of discrete events occurring in the process. Figure 2 explains how this system can be represented in terms of discrete event model. The process consists of the customer arrival event, Order processing event, Order preparation event and Order delivery event. There are delay variables for each section of the system namely. There are further delays specified for each section defined separately. We may need to define more variable in the system to calculate its efficiency. Currently we are considering the delay times to be the efficiency parameter. The simulation will yield a result giving the average service time per order and specific time for each order and we try to minimize it by adjusting the parameters. The cooking times for each specific item in the kitchen are constants. The customer can order specific combination of orders and can also order more than one item at a time. This leads to variation in service time and there needs a proper simulation to test it.

## Part 2-Drive Thru Service Model Components and Test Strategy

2.1 Atomic Model-Ordering Window

The general formulation of atomic model is given by:

M = <*X, Y, S, δint, δext, λ, ta*>

X={(IN_ORDER, ***N***)};

*S* = {Ordertype, Fryer,Pans,Salad,Grill};

*Y* = {(F_1, ***N***); (P_1, ***N***); (S_1, ***N***); (G_1, ***N***)};

Transistion Functions :
if (IN_ORDER=1)
**{**
***δext = Salads;***

***preparation time=delay time in loop***

***δint (S_1,1)***

}

if (IN_ORDER=2)
**{**
***δext = Fryers;***

***preparation time= delay time in loop***

***δint (F_1,1)***

}

if (IN_ORDER=3)
**{**
***δext = Grill;***

***preparation time= delay time in loop***

***δint (G_1,1)***

}

if (IN_ORDER=4)
**{**
***δext = Pans;***

***preparation time=00:00:00:11***

***δint (P_1,1)***

}

Else

{do nothing}

2.2 Coupled Model-Kitchen

*M = <X, Y, D, {Md}, EIC, EOC, IC>*

X={S_1,F_1,G_1,P_1}

Y={OUT}

D={PREP_TIME,PANS,SALAD,GRILL,FRYER,F_P,P_P,S_P,G_P}

*EIC=NULL*

*EOC=NULL*

*IC={* ( (FRYER, out), (F_P, in)); ( (F_P, out), (FRYER, done)); ( (PANS, out), (P_P, in)); ( (P_P, out), (PANS, done)); ( (GRILL, out), (G_P, in)); ( (G_P, out), (GRILL, done)); ( (SALAD, out), (S_P, in)); ( (S_P, out), (SALAD, done));
*; }*

**Part 3-Implementation of the Model**

The structure of the model shows atomic and coupled parts of the model.

There are three parts to the design

1. Order window (Atomic)
2. Kitchen(Coupled)
3. Order Packing(Atomic)

The order window is fed with the inputs of customers.ev which gives the input of tiem number in the menu. For simplicity of the work we have taken 4 items in the menu.The tiems are categorized 4 types

1. Salad Bowl (Works on the Salad function in kitchen_procType.cpp)
2. Fried Chicken and French Fries (Works on the fryer function in kitchen_procType.cpp)
3. Grilled Turkey (Works on the Grill function in kitchen_procType.cpp)

4. Pancake (Works on the Pans function in kitchen_procType.cpp)

The inputs file randomly input the value the orders at the order window. The orders are taken at the order window and segregated at the output that goes to the kitchen model. The output from the atomic model of order window is done with the help of a

The Kitchen block is designed to handle different orders and work with the inputs from the atomic window block. There are 4 inputs for all the section of the kitchen. Each input lead to a function that processes the order in a specified time. Random distribution of time consumption could have been taken into consideration but for simplicity of the work each function of the kitchen is assigned a specific time to finish the order.

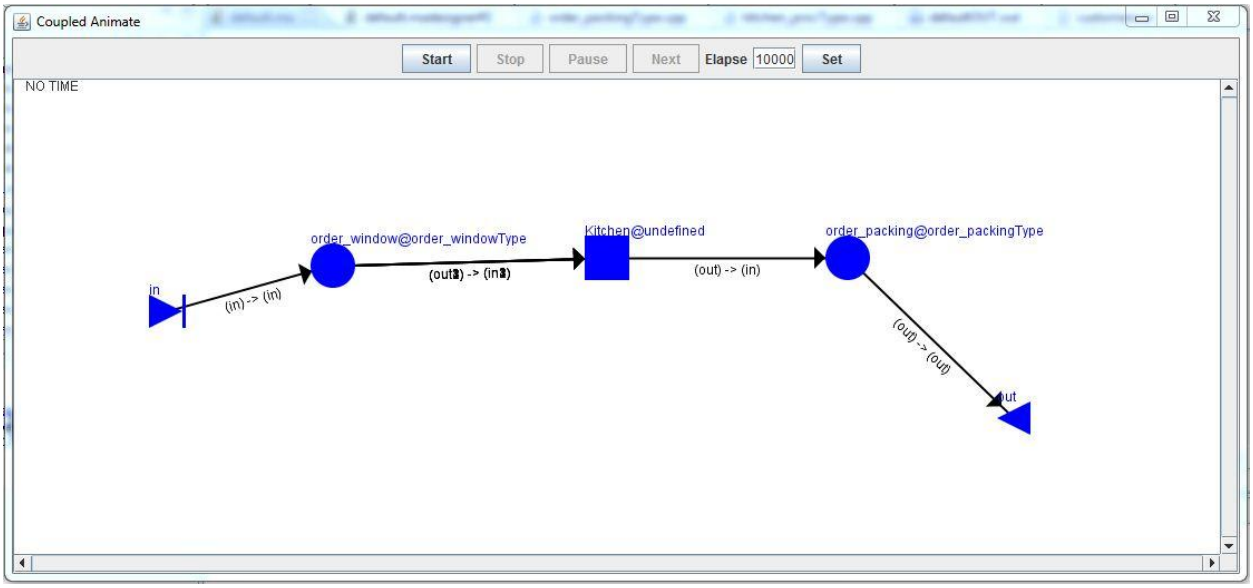
Each input for the Kitchen coupled model is monitored to evaluate the time of processing the order. There are 4 parts for the inputs going to salads, fryer, grill and pans.

Once the orders are done we need to pack them to give to the customer, for this we can see there is order packing window which takes input from the kitchen block and uses a uniform time to pack the food.

The running instructions for the project is written in the file named Runnig_instruction.txt in the project folder.

**Results**

As we run the simulation with customers.ev file we see the following figure in the simulation window which changes with time:



We see the changes in the states as the customer.ev file inputs the value. The outputs are visible at the out port. As we can visualize with the simulation that the orders mover in the respective sections and get processed by the routine. The output are as expected of the system and the simulation yields the required results.