**Workshop seminar 3**

**Comp201 Software Engineering**

**Answers**

Q1. What are the conditions needed for a Petri net to be deterministic?

Only 1 transition must be enabled at once.

Q2 What are the conditions needed for a Petri net to model a finite state system?

It must have a finite reachability tree.

Q3. In lecture’s 9 and 10, slide 26, we had a petri net model for email sending/receiving.

receive\_mail

P1

begin

typing\_mail

rest

rest

read\_mail

ready

P2

send\_mail

Here is a solution for dealing with the problem of a limited buffer size.

What do the tokens in P1 represent? P1 represent the numbers of email messages, at the mail server but not downloaded.

What do the tokens in P2 represent?

This represents the amount of free space in the mail server;s buffer, every time a new mail is sent this goes down by one. If P2 has zero tokens you cannot send any more mail until a new mail is downloaded.

Q4 The next model is a supply chain and manufacturing example, for manufacturing a product out of 3 separate components A, B and C. What could the places, tokens and transitions represents.

What timing constraints would be useful?

**T1  
Make  
A**

4

**T4  
Send A**

**Make  
product**

**T2**

**T5**

**T6**

**T3**

Tokens in PA1, PA2, PA3 these would be outstanding orders for the components.  
T1,T2, T3 represents the manufacturing of the components. The output of these represents storage at the component manufactures. The timings on these represent the range of times it will take to make each component. T4,T5 and T6 represents the sending of the components, timing here represents how long transportation is (fast by plane, slow and cheap by ship)

How would be change the model so that 4 x components A as needed to make the product.

Add in weight of 4 between PA storage and T7. See diagram.

How could you add to the model to make sure that a new order is sent for 200 components every time the number of components in the factory warehouse falls to less than 100.

**T4  
Send A**

**T7**

200

**Update  
level  
maxtime=0**

**Send order to**

**Supplier**

**MaxTime=1**

100

**Figure 1**

In this case, the flow starts with an order, once that order has been made, the system will wait until the item’s arrive in storage. It will start of by ordering 200 items. This will remove 100 out of the components needed. As components arrive and are marked as delivered this updates the order control place. Every time a product is made this updates the components needed place, the update level transition is used to remove components from the components needed place for outstanding arrivals.

We might want to model with an alternate supplier for one of the components, how would we do this.

**T4.1**

**T1.1**

**T1.2**

**T4.2**

Now imagine the component supplier 2 is lot more expensive but has a guaranteed 1/day delivery, supplier but supplier 1 is half the price but has a possibly longer leader time.

How would could you extend the model to keep having enough of produce A but not pay

too high a price.

Answer duplicate the logic in Figure 1, x 100 for fast supplier, x 1000 for slow supplier.  
the T4.1 and T4.2 would both have outputs to the stock control inputs so that the arrival of a component from either supplier would update the level needed.

**Update**

**2000**

**Order 2000**

**2000**

**Make  
product**

**T4.1**

**T1.1**

**T1.2**

**T4.2**

**100**

**Order 100**

**100**

**Update**

Q5. What is the problem with these 2 petri nets with timing constraints?

**T1  
tmin=1  
tmax=1**

**T2  
tmin=1  
tmax=1**

This petri net would not be atomic, both transitions would fire at the same time, this is not allowed and there would be -1 token left in the place.

**T2  
tmin=0  
tmax=0**

**T1  
tmin=0  
tmax=0**

There would be an infinite number of tokens and both transitions would fire at the same time.