Exercise & Case study (Telecom Systems)

F. Kordon & C. Girault - MATCH School

Installation - if you did not run Exercise 1 on your account.

First, you must get all the information (models for this exercise and formalisms) to be operational. To do so, please type first (when logged under Unix) the following commands :

```
>cd
>cp -r /export/home/profesores/kordon/MACAO .
```

Then, typing “ls”, you notice a MACAO folder in your home directory. All required information is there. When you run Macao under your Home Directory and open models, You will get into this MACAO that contains two folders :

- FORMALISMS that contains the AMI-Net description (you go there when you create a new model),
- MODELS in which we will insert (if required) directories containing models for exercises.

Installation - if you did run Exercise 1 on your account.

First, you must get all the information (models for this exercise and formalisms) to be operational. To do so, please type first (when logged under Unix) the following commands :

```
>cd
>cd MACAO/MODELS
>cp -r /export/home/profesores/kordon/MACAO/MODELS/EXERCISE_TELECOM .
>cp -r /export/home/profesores/kordon/MACAO/MODELS/SOLUTION_TELECOM .
```

All should read this.

1) In the MACAO/MODELS folder, you will find the EXERCISE_TELECOM folder containing two models. File identifier starts with a number that corresponds to the question you are required to use the model.

2) In the MACAO/MODELS folder, you will find the SOLUTION_TELECOM folder containing modeling solutions. If you want to check your solution or get tired of looking for it, you can get them. File identifier starts with a number that corresponds to the related question.

3) There is an annex (at the end of this document) that provides you with some basics about PROD queries (menu «Evaluation of the RG with PROD/Expert mode/Build a query...» in CPN-AMI).

1. Design of a safe channel

We would like to design a safe channel based on a single cable line. The usual problem with a unique cable is that electric signals coming from various origins may provoke collisions (message is lost). To ensure a safe communication on the channel, we propose the architecture of Figure 1.

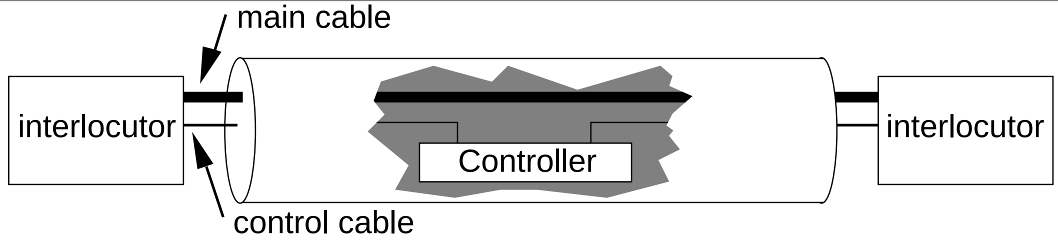

Figure 1: Structure of a line.

The channel relates two *interlocutors* that communicate together. It is composed of a *line* and a *controller* that manages shared access to the channel *main cable* (128 bits width). The controller is connected two each interlocutor with a discrete *control cable*

(3 bits width). There is one control cable per interlocutor. Interlocutors cannot send a signal at the same time : they must ask first the line to the controller that accepts or refuse (according to an implemented strategy).

Interlocutors have to respect the following protocol :

(1) the default state for an interlocutor is listening to the main cable,
(2) when it wants to emit a signal, the interlocutor asks for the main cable,
(3) if the controller provides the main line, then, the interlocutor sends its message and waits for an acknowledge,
(4) if the controller refuses the main line, then the interlocutor listens to the main cable (a message should arrive) and retries later on,
(5) interlocutors only send one message at a time,
(6) when an interlocutor gets its acknowledge, it frees the line for another use,
(7) Only messages passing through the main cable are acknowledged[1].

The table above provides the identification of signals passing through the cables.

Signal name	Signification	Signal direction[2] Interlocutor — Controller (between interlocutors)	Signal direction[2] Interlocutor — Controller (control cable)
AMC	Ask for main cable		►
RMC	Refuse main cable		◄
PMC	Provide main cable		◄
MSG	Message	◄ ►	
ACK	Acknowledge	◄ ►	
FMC	Free the main cable		►

Signal between two interlocutors are transmitted using the main cable. Other signals are transmitted on the control cables.

Some typical execution scenarii are provided hereafter to illustrate the behavior of a interlocutor according to specific situations.

Figure 2 illustrates the behavior of a interlocutor that initiates a communication when the controller provides the main cable. Then, the answer to AMC (demand) is PMC. The interlocutor (here, 1) then sends the message to the other interlocutor (here, 2) and waits for an acknowledge. When it gets the acknowledge, it releases the main cable (FMC).

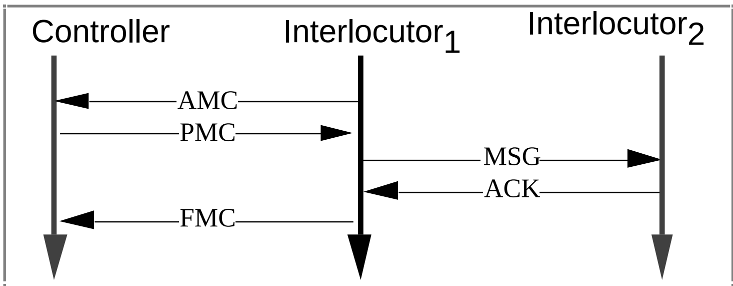

Figure 2: OMT-like scenario of an accepted connection.

Figure 3 shows the behavior of an interlocutor when the controller refuses the main cable (RMC). It means that the other interlocutor (here, 2) obtained the cable and is sending data. The interlocutor (here, 1) then waits for the incoming message and acknowledges it. The interlocutor will try later to get the main cable.

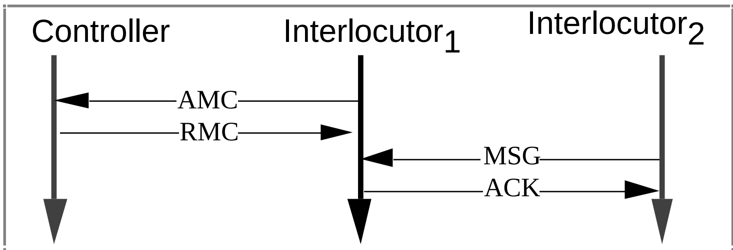

Figure 3: OMT-like scenario of a refused connection.

1 Acknowledgemen will be usefull further in the study.
2 ◄ and ► provides the direction of the signal, ◄ ► means that the signal is exchanged between two interlocutors.

The aim of this study is to model this system (the controller and the interlocutors) with P/T nets in order to validate it. To do so, we model separately the behavior of each components of the safe channel. Then, we connect them together to study the system.

One modeling technique is to let Petri net modules communicate by means of communication places. Signal identified in the table are a good candidate for these communication places. We thus identify six places that will act as interfaces between the two components of the system : AMC, RMC, PMC, MSG, ACK and FMC. The signification of these places is "some signal is in" (signal AMC for place AMC, etc.).

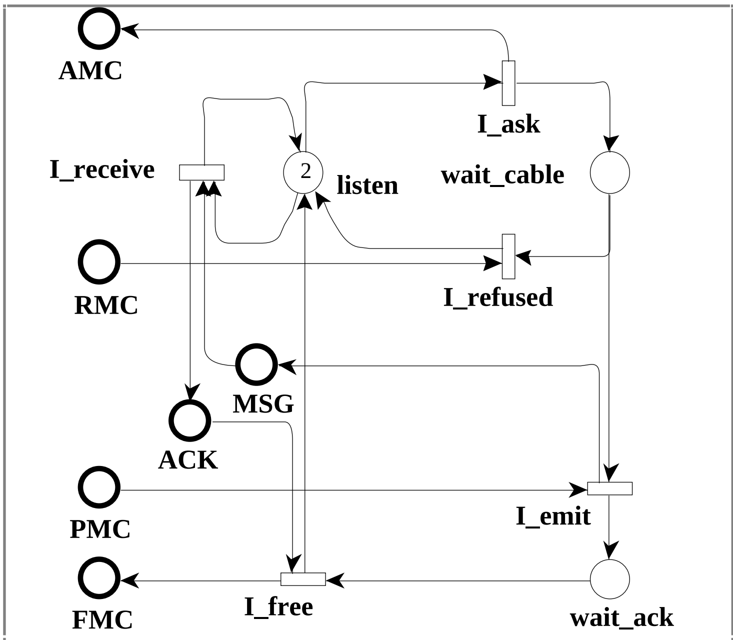

Figure 4: Petri net modeling interlocutors in the system.

We then provide the Petri net that models interlocutors in the system (Figure 4). Interfaces places are bolded. Each interlocutor listen to the main cable (place *listen*). When it asks for the cable (transition *I_ask*), it may get either a RMC (and then goes back to *listen*) or a PMC. It then emit the message and waits for its acknowledge. When it comes, the interlocutor frees the line (transition *I_free*). There are two interlocutors (on each end of the channel) and thus two tokens[3] in *listen*.

Question 1

Complete the Petri net model by inserting the behavior of the controller.

Question 2

Check the T-invariants with GreatSPN services (under CPN-AMI) on the complete model. Can you provide a firing sequence and an interpretation for them?

Question 3

Check the P-invariants with GreatSPN services on the complete model. Some of them can be interpreted. Can you provide an interpretation?

Question 4

Subtract P-invariants two to P-invariant three. What can you deduce? do you observe something similar with others couple of invariants?

Question 5

Is the net bounded? use the corresponding tool to check it.

Question 6

Some properties are stated on the Safe Channel. They are listed below :

- **P1** There is only one electric signal at a time in the main cable.
- **P2** There can never be two interlocutors emitting in the main cable at the same time.
- **P3** If the interlocutors do not crash elsewhere, the protocol between components of the channel does not introduce communication problems.

3 This works only because there are two interlocutor.

Translate these textual properties in terms of Petri net properties to verify.

Question 7

What formal properties can you use to validate properties P1 and P2?

Question 8

Generate the reachability graph. Using model checking techniques (with PROD), try to verify the corresponding properties. You should use the expert mode and type your queries using the information provided in annex.

Question 9

You can have a look on the Reachability graph if you want.

2. Managing loss in the main cable

It appears that the 128 bit width main cable is not as safe as the 3 bits width control cables. Thus, the electric signal can be loss (i.e. not interpreted as a message by a interlocutor). It is thus decided to introduce a time-out. After that time-out, the emitting interlocutor will send again his message.

Question 10

By adding one transition in the previous P/T net, model the fact that an interlocutor can re-emit a message in the main cable. So far, only consider the loss of messages (MSG signal).

Question 11

Compute bounds of this model. What effect do you observe?

Question 12

What do you deduce about the reachability graph?

Question 13

Where is the problem, Modify the P/T net model to solve it. Please consider now all loss on the main cable (MSG as well as ACK). Like in TCP/IP, the emiter re-emit the message when te Acknowledge is lots. The receiver then handles it.

Important: *we assume that there is not "bad time-out" (i.e. a time-out is generated only if there is a message loss).*

Question 14

Use PROD queries to list states of the reachability graph that correspond to a time-out generation. How many states do you identify?

Question 15

For each identified state, use PROD queries to get a scenario that leads to the generation of a time-out.

Question 16

To check if the protocol is safe despite message loss, it is stated that when a message is sent, an acknowledge has to be received. Can you express that using a PROD query and run it? Is this property respected?

Question 17

Somebody suggests the problem comes from message loss that could generate infinite repetitive sequences of particular transitions. How can you detect such a sequence using structural properties? run the appropriate tool to demonstrate this idea

Question 18

Can you provide an interpretation of these two invariants? what hypothesis can we reasonably suggest on the main cable?

Question 19

Modify the Petri net model to handle that new hypothesis

Question 20

Process the query of question 16. Is it now verified? Can you use a similar way to the one of question 17 to find an explanation?

Hint*: have a look on the behavior of interlocutors.*

Question 21

Does that comes from the modeling of message loss on the main cable? Can you propose a modification on the previous model?

Hint: *you have to change the behavior of interlocutors by preventing this infinite loop.*

Question 22

Use PROD to process the query of questions 16 and 20. Is it now verified?

Question 23

Use now PROD to display statistics on the reachability graph What particular states can you observe?

Question 24

Use PROD to identify path that lead to these particular configurations and propose a solution to the problem.

Question 25

Modify the model in order to take into solve this problem.

Question 26

Check the bounds of places with the appropriate service in CPN-AMI. Do you observe something suspicious? if yes, check with PROD and propose an explanation.

3. Designing a bus

Having successfully designed the safe channel, we would like to use a similar strategy to design a safe bus that can connect up to N users. The architecture of this bus is similar to the one presented in section 1. Each interlocutor is connected to the controller using a dedicated control cable and have a unique identification number. Each one is also connected to the maim cable. When an interlocutor sends a message, it provides the identification of its corespondent. Acknowledge does not requires identification while only the sender is listening to it in the main cable. The model is similar to the one of the safe channel.

The Petri net of Figure 5 models such a behavior. Interlocutors are identified using the *It* class. To evaluate the design, we set the maximum of connected interlocutors to 4.

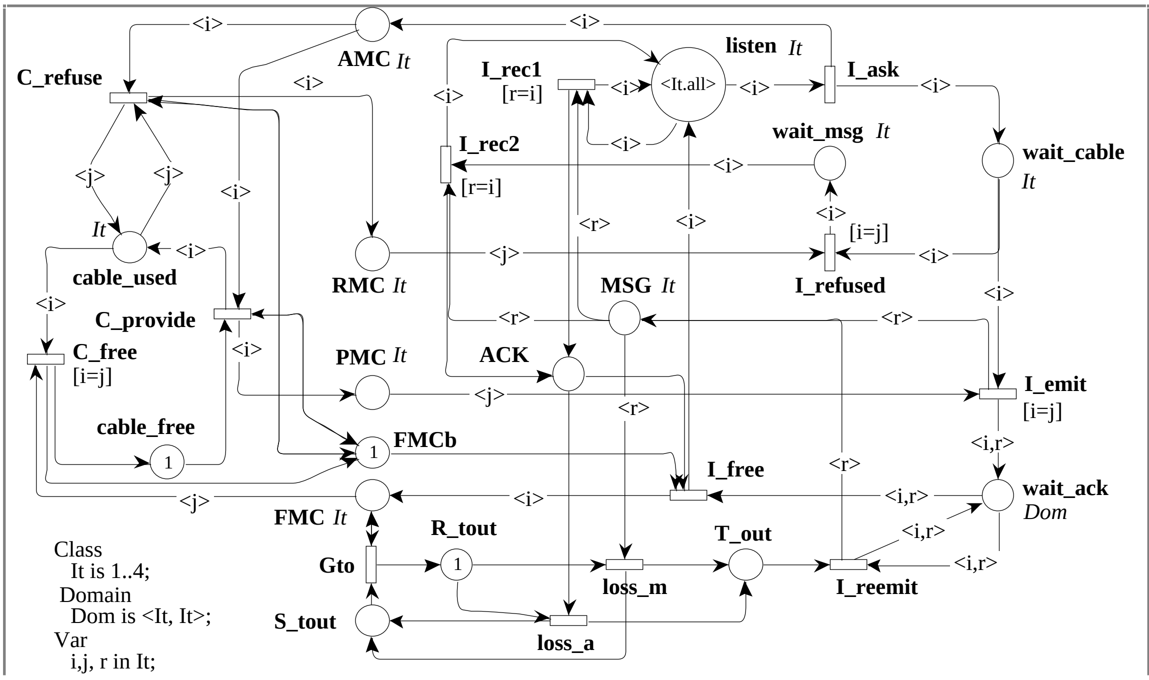

Figure 5: Petri net modeling the controller and four interlocutors.

Question 27

Generate the reachability graph for this model and have a look on the statistics (do not think one second at downloading it;-). Is the Petri net deadlock-free?

Question 28

Use the tools to propose an explanation to what you observed in the previous question.

Question 29

Propose a correction that solves the problem.

Question 30

How can you express in PROD that, if a client <i> sends a message to a given client j ≠ i, it has to get an acknowledge.

Question 31

Somebody is asking about the fairness of the protocol : it means that no interlocutor gets into starvation. Use PROD to demonstrate that it can happens.

Question 32

It is suggested that interlocutors are not correctly modeled. When a request for the main cable is refused, it should systematically try to re-emit it (it may currently change its mind, which appears to be strange). Change the model to take into consideration this new aspect and apply the same query to prove that this protocol is really not fair.

Question 33

To be fair, the controller must sort demands. Model this new controller behavior and apply the previous query to check the fairness of the system.

Annex :

some information about the way to express PROD commands

The table above summarize some useful queries to play with PROD (expert mode). Please remind that all nodes in the reachability graph are numbered. The initial marking is node #0. When PROD results are displayed on the historic window, you get the identification number of the corresponding states in the reachability graph.

The “verbose” option provides you with detailed information. If you omit it, you will only get the identification number of nodes in the reachability graph.

PROD Query	Meaning
`query [verbose] node ($1 = $2)`	provides all the states in the reachability graph for which place **$1** has marking **$2** . Remind that a non colored token is noted <..> in prod.
`query [verbose]node (card($1) > $2)`	provides all the states in the reachability graph for which place **$1** has more than **$2** token(s). You can also use other operations (<, ==, >=, <=, !=).

`goto $1`
`query true`
`goto $2`
`query [verbose]bspan (true) %1`
	provides the shortest state between nodes **$2** and **$1** in the reachability graph.

`query [verbose]node ((c1) and (c2))`	provides all the states in the reachability graph that respect conditions **c1** and condition **c2**. You can do the same with or.
`query [verbose]AG`[4] `(IfThen ($1, AF`[5] `($2)))`	Are all states in the reachability graph that respect condition **$1** eventually leads a states that respect condition **$2**.
`look $1`	Display the marking corresponding to node **$1** in the reachability graph.

Tokens in PROD are represented as follow :

- <..> for non colored tokens
- <.x.> where x is a value for colored tokens
- <.x_1,...,x_n.> for composed colored tokens (x_i are values of the appropriate color classes)

4 AG can also be written HenceforthOnAllBranches in some PROD configurations.
5 AF can also be writtent EventuallyOnAllBranches in some PROD configurations.