SRML Editor Tutorial (4)

CO7205 Advanced System Design

José Fiadeiro
Laura Bocchi
Yi Hong

Lab sheet and materials

http://www.cs.le.ac.uk/srml/

Wires and Interaction Protocol

Your task:

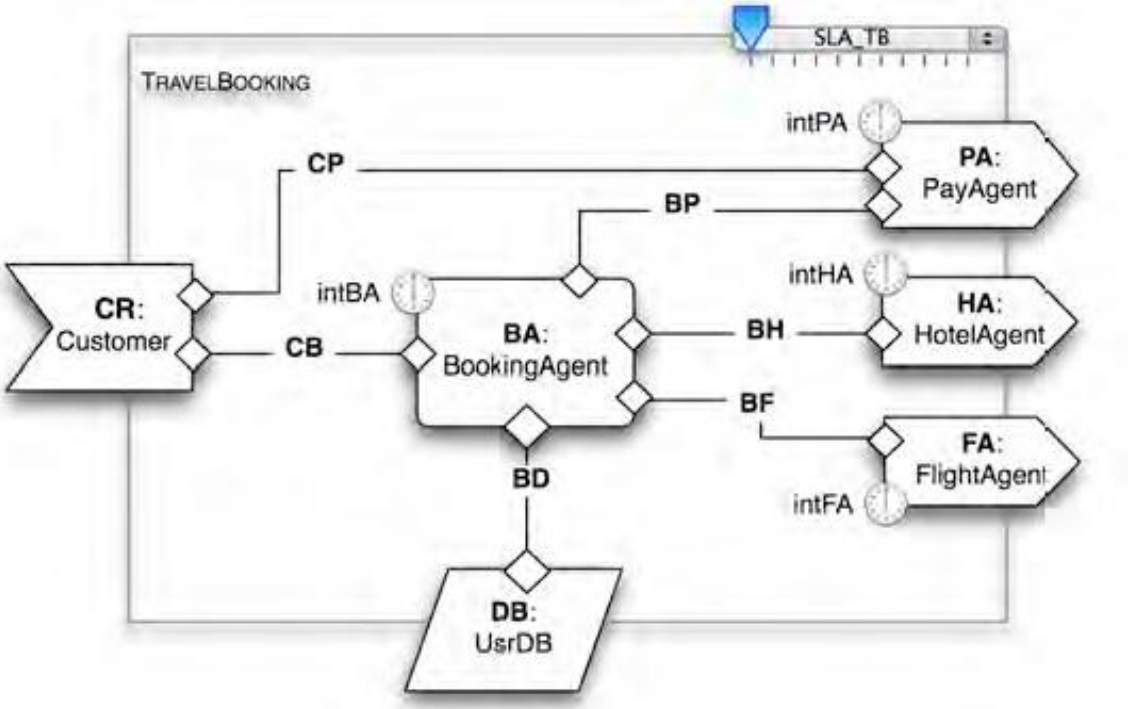
Complete interaction protocols and wires of TravelBooking example

(1)Select ***File->New->Other*** from eclipse menu to open a project wizard,
(2)type ***SRML*** to shortlist the options
(3)select ***srml Project*** then click "***Finish***".
(4)Download SRML source code of "***TravelBooking***" from:
http://www.cs.le.ac.uk/srml/example/travelBooking_incompleted.txt
(5) Copy and paste all content of Procurement_incomplete.txtto model.srml

* *TravelBooking example can be found in the appendix of SRM Lecturer note (Page 54).*

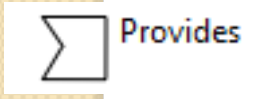
Wires and Interaction Protocol

TravelBooking



TravelBooking: Compoment & Interface

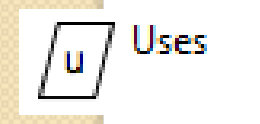
TravelBooking example consists of:



• CR – the provides-interface of the module, of type *Customer;*

• FA – a requires-interface (for a flight-booking service), of type *FlightAgent;*
• PA – a requires-interface (for a payment service), of type *PayAgent;*
• HA – a requires-interface (for a hotel-booking service), of type *HotelAgent;*

• BA – an interface for a component that coordinates the business process, of type *BookingAgent;*



• DB – a uses-interface for a persistent component (of the bottom layer) that stores user data, of type *UsrDB;*

• *CB, CP, BF, BH, BP, BD – wire-interfaces typed by connectors that establish* the required interconnection

Interaction Protocol

Syntax of Interaction Protocol in the editor:

```
INTERACTION PROTOCOL i/o is
    ROLE A
        ask A1(d1):d2
    ROLE B
        rpl R1(d1):d2
    COORDINATION
        (x:d1) A1(x) = R1(x)
```

datatype AbstractTypes is
sort d1;
sort d2;
endd

interaction protocol IO is

imports AbstractTypes;

Define and import data types when necessary

roleA ask A
param1 : d1,
return param2 : d2 ;

roleB rpl R
param1 : d1,
return param2 : d2;

endip

`ask A1(d1):d2` → **ask A**
param1 : d1,
return param2 **:** d2**;**

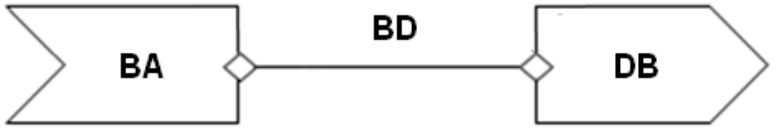
Note: Coordination is currently not supported in the editor

Wires

Syntax of SRML wire in the editor:

Example: Wire ***BD***

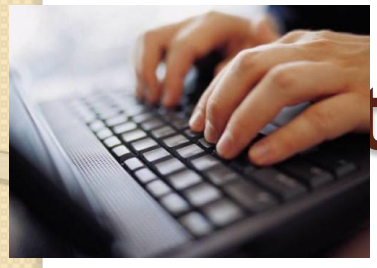
(Page 57 of Lecturer Note)




```
BD {
nodeA BA;
nodeB DB;
   connector {
      attachmentA { log=>A; }
      attachmentB { log=>A; }
    interaction protocol IO;
   }
   connector {
      attachmentA { getData=>A; }
      attachmentB { getData=>A; }
    interaction protocol IO; }
   connector {
      attachmentA { getCard=>A; }
      attachmentB { getCard=>A; }
    interaction protocol IO;
   }
}
```

BA BookingAgent	c_6	BD	d_6	DB UserDB
ask log	A_1	i/o	R_1	rpl log
ask getData	A_1	i/o	R_1	rpl getData
ask getCard	A_1	i/o	R_1	rpl getCard

Interaction Protocol IO
(refer to previous page)



tices

Tips:

Make use of
Ctrl+ space
(auto completion)

Exercise:

Complete Wires ***BF/CP/BP/BH***
(page 56, 57 of SRML Lecturer Note)
and relevant Interaction protocols

BA BookingAgent	c_4	BF	d_4	FA FlightAgent
s&r bookFlight	s_1	Straight. I(airport, airport, date, date, usrdata) O(fcode, moneyvalue, accountn, serviceId)	R_1	r&s lockFlight
🔔 from	i_1		i_1	🔔 from
to	i_2		i_2	to
out	i_3		i_3	out
in	i_4		i_4	in
traveller	i_5		i_5	traveller
✉ fconf	o_1		o_1	✉ fconf
amount	o_2		o_2	amount
beneficiary	o_3		o_3	beneficiary
payService	o_4		o_4	payService
snd payAck	s_1	Straight. I(pcode, bool)	R_1	rcv payAck
🔔 proof	i_1		i_1	🔔 proof
status	i_2		i_2	status
rcv ackRefundRcv	R	Straight. I(moneyvalue)	S	snd payRefund
🔔 amount	i_1		i_1	🔔 amount

CR Customer	c_5	CP	d_5	PA PayAgent
rcv payNotify	R_1	Straight. I(bool)	S_1	snd payNotify
🔔 status	i_1		i_1	🔔 status

BA BookingAgent	c_2	BP	d_2	PA PayAgent
s&r payment	S_1	Straight. I(moneyval, accountn, usrdata, paydata) O(pcode)	R_1	r&s payment
🔔 amount	i_1		i_1	🔔 amount
beneficiary	i_2		i_2	beneficiary
originator	i_3		i_3	originator
cardNo	i_4		i_4	cardNo
✉ proof	O_1		O_1	✉ proof

BA BookingAgent	c_3	BH	d_3	HA HotelAgent
s&r bookHotel	S_1	Straight. I(date, date, usrdata) O(hcode)	R_1	r&s lockHotel
🔔 checkin	i_1		i_1	🔔 checkin
checkout	i_2		i_2	checkout
traveller	i_3		i_3	name
✉ hconf	O_1		O_1	✉ hconf