To simplify the use of JTT, we present a complete example running under this tool. This example illustrates the case that we have 1 Merchant and 3 Customers. The tool starts by asking the name of the model, its instance(s) and states.

```
***Java Tranformer Tool Started***

Please insert the Name of the model (# to end):
Customer
Please enter the name of each instance of this model and press enter after each name:
(# to end)
cus1
cus2
cus3
#
Please enter the states of the model and press enter after each state[]: (# to end)
c0
Please enter the states of the model and press enter after each state[c0]: (# to end)
c1
Please enter the states of the model and press enter after each state[c0, c1]: (# to
end)
c2
Please enter the states of the model and press enter after each state[c0, c1, c2]: (#
to end)
c3
Please enter the states of the model and press enter after each state[c3, c0, c1,
c2]: (# to end)
c4
Please enter the states of the model and press enter after each state[c3, c4, c0, c1,
c2]: (# to end)
c5
Please enter the states of the model and press enter after each state[c3, c4, c5, c0,
c1, c2]: (# to end)
c6
Please enter the states of the model and press enter after each state[c3, c4, c5, c6,
c0, c1, c2]: (# to end)
c7
Please enter the states of the model and press enter after each state[c3, c4, c5, c6,
c7, c0, c1, c2]: (# to end)
c8
Please enter the states of the model and press enter after each state[c3, c4, c5, c6,
c7, c8, c0, c1, c2]: (# to end)
c9
Please enter the states of the model and press enter after each state[c3, c4, c5, c6,
c7, c8, c9, c0, c1, c2]: (# to end)
#
Please enter the initial state of the model[c3, c4, c5, c6, c7, c8, c9, c0, c1, c2]:
c0
Do you have a commitment in state c4 (yes/no)?
no
From state c4, enter the actions and press enter after each, press # to end
Null
```

```
For state c4, Action Null, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state c4 with Action Null, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c0
From state c4, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state c5 (yes/no)?
no
From state c5, enter the actions and press enter after each, press # to end
Deliver
For state c5, Action Deliver, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
#
From state c5 with Action Deliver, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c7
From state c5, enter the actions and press enter after each, press # to end
notDeliver
For state c5, Action notDeliver, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
#
From state c5 with Action notDeliver, enter the target state[c3, c4, c5, c6, c7, c8,
c9, c0, c1, c2]:
c6
From state c5, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state c6 (yes/no)?
no
From state c6, enter the actions and press enter after each, press # to end
Refund
For state c6, Action Refund, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
#
From state c6 with Action Refund, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c8
From state c6, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state c7 (yes/no)?
no
From state c7, enter the actions and press enter after each, press # to end
Receipt
For state c7, Action Receipt, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
```

```
#
From state c7 with Action Receipt, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c9
From state c7, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state c8 (yes/no)?
no
From state c8, enter the actions and press enter after each, press # to end
Null
For state c8, Action Null, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state c8 with Action Null, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c0
From state c8, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state c9 (yes/no)?
no
From state c9, enter the actions and press enter after each, press # to end
Null
For state c9, Action Null, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state c9 with Action Null, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c0
From state c9, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state c0 (yes/no)?
no
From state c0, enter the actions and press enter after each, press # to end
Request
For state c0, Action Request, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state c0 with Action Request, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c1
From state c0, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state c1 (yes/no)?
no
From state c1, enter the actions and press enter after each, press # to end
Quote
For state c1, Action Quote, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
#
```

```
From state c1 with Action Quote, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c2
From state c1, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state c2 (yes/no)?
no
From state c2, enter the actions and press enter after each, press # to end
Reject
For state c2, Action Reject, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state c2 with Action Reject, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c4
From state c2, enter the actions and press enter after each, press # to end
Accept
For state c2, Action Accept, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state c2 with Action Accept, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c3
From state c2, enter the actions and press enter after each, press # to end
#
```

If a state has commitment then the user should insert where this commitment is fulfilled

```
Do you have a commitment in state c3 (yes/no)?
yes
What is the state where this commitment is fulfilled[c3, c4, c5, c6, c7, c8, c9, c0,
c1, c2]?
c5
From state c3, enter the actions and press enter after each, press # to end
notPayment
For state c3, Action notPayment, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state c3 with Action notPayment, enter the target state[c3, c4, c5, c6, c7, c8,
c9, c0, c1, c2]:
c4
From state c3, enter the actions and press enter after each, press # to end
Payment
For state c3, Action Payment, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state c3 with Action Payment, enter the target state[c3, c4, c5, c6, c7, c8, c9,
c0, c1, c2]:
c5
From state c3, enter the actions and press enter after each, press # to end
#
```

After that, user has to insert the atomic propositions of this model

```
Please enter module atomic proposition (# to end):
Pay
Please enter the instances of this atomic proposition (press # to end):
cus1
cus2
cus3
#
Please enter the state of this atomic proposition[c3, c4, c5, c6, c7, c8, c9, c0, c1,
c2]:
c5
Please enter module atomic proposition (# to end):
Deliver
Please enter the instances of this atomic proposition (press # to end):
cus1
cus2
cus3
#
Please enter the state of this atomic proposition[c3, c4, c5, c6, c7, c8, c9, c0, c1,
c2]:
c7
Please enter module atomic proposition (# to end):
#
```

This process continues until inserting all information for all states in the first model. Hereafter, the user can start inserting the second model information.

```
Please insert the Name of the model (# to end):
Merchant
Please enter the name of each instance of this model and press enter after each name:
(# to end)
mer1
#
Please enter the states of the model and press enter after each state[]: (# to end)
m0
Please enter the states of the model and press enter after each state[m0]: (# to end)
m1
Please enter the states of the model and press enter after each state[m0, m1]: (# to
end)
m2
Please enter the states of the model and press enter after each state[m0, m1, m2]: (#
to end)
m3
Please enter the states of the model and press enter after each state[m0, m1, m2,
m3]: (# to end)
m4
Please enter the states of the model and press enter after each state[m0, m1, m2, m3,
m4]: (# to end)
m5
Please enter the states of the model and press enter after each state[m0, m1, m2, m3,
m4, m5]: (# to end)
m6
```

```
Please enter the states of the model and press enter after each state[m0, m1, m2, m3,
m4, m5, m6]: (# to end)
m7
Please enter the states of the model and press enter after each state[m0, m1, m2, m3,
m4, m5, m6, m7]: (# to end)
m8
Please enter the states of the model and press enter after each state[m0, m1, m2, m3,
m4, m5, m6, m7, m8]: (# to end)
m9
Please enter the states of the model and press enter after each state[m0, m1, m2, m3,
m4, m5, m6, m7, m8, m9]: (# to end)
#
Please enter the initial state of the model[m0, m1, m2, m3, m4, m5, m6, m7, m8, m9]:
m0
Do you have a commitment in state m0 (yes/no)?
no
From state m0, enter the actions and press enter after each, press # to end
Request
For state m0, Action Request, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
arg3
arg4
#
From state m0 with Action Request, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m1
From state m0, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state m1 (yes/no)?
no
From state m1, enter the actions and press enter after each, press # to end
Quote
For state m1, Action Quote, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state m1 with Action Quote, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m2
From state m1, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state m2 (yes/no)?
no
From state m2, enter the actions and press enter after each, press # to end
Reject
For state m2, Action Reject, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
arg3
arg4
#
```

```
From state m2 with Action Reject, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m4
From state m2, enter the actions and press enter after each, press # to end
Accept
For state m2, Action Accept, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
arg3
arg4
#
From state m2 with Action Accept, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m3
From state m2, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state m3 (yes/no)?
no
From state m3, enter the actions and press enter after each, press # to end
notPayment
For state m3, Action notPayment, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
arg3
arg4
#
From state m3 with Action notPayment, enter the target state[m0, m1, m2, m3, m4, m5,
m6, m7, m8, m9]:
m4
From state m3, enter the actions and press enter after each, press # to end
Payment
For state m3, Action Payment, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg2
arg3
arg4
#
From state m3 with Action Payment, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m5
From state m3, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state m4 (yes/no)?
no
From state m4, enter the actions and press enter after each, press # to end
Null
For state m4, Action Null, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state m4 with Action Null, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
```

```
m0
From state m4, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state m5 (yes/no)?
yes
What is the state where this commitment is fulfilled[m0, m1, m2, m3, m4, m5, m6, m7,
m8, m9]?
m7
From state m5, enter the actions and press enter after each, press # to end
Deliver
For state m5, Action Deliver, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state m5 with Action Deliver, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m7
From state m5, enter the actions and press enter after each, press # to end
notDeliver
For state m5, Action notDeliver, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state m5 with Action notDeliver, enter the target state[m0, m1, m2, m3, m4, m5,
m6, m7, m8, m9]:
m6
From state m5, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state m6 (yes/no)?
no
From state m6, enter the actions and press enter after each, press # to end
Refund
For state m6, Action Refund, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state m6 with Action Refund, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m8
From state m6, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state m7 (yes/no)?
no
From state m7, enter the actions and press enter after each, press # to end
Receipt
For state m7, Action Receipt, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state m7 with Action Receipt, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m9
From state m7, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state m8 (yes/no)?
```

```
no
From state m8, enter the actions and press enter after each, press # to end
Null
For state m8, Action Null, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state m8 with Action Null, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m0
From state m8, enter the actions and press enter after each, press # to end
#
Do you have a commitment in state m9 (yes/no)?
no
From state m9, enter the actions and press enter after each, press # to end
Null
For state m9, Action Null, who is performing this action (if the current agent
performs the action, enter arg1; otherwise, insert the name of the agent
(arg2,...,argn), # to end):
arg1
From state m9 with Action Null, enter the target state[m0, m1, m2, m3, m4, m5, m6,
m7, m8, m9]:
m0
From state m9, enter the actions and press enter after each, press # to end
#
Please enter module atomic proposition (# to end):
#
Please insert the Name of the model (# to end):
#
```

After that, the user has two choices to insert the specifications; either using console, or using an input text file. In the figure below, the user inserts the specifications through the input file (formula.txt).

```
Specifictions, do you want to enter specs from console or have them read them from a
file (file,console)?
file
Please enter input filename (default: formula.txt)?
formula.txt
```

When user finishes inserting the specifications, the program will display the output extended NuSMV code and store it in output.txt file.

```
Module main
Var
cus1 : process <Customer>(cus1,mer1);
cus2 : process <Customer>(cus2,mer1);
cus3 : process <Customer>(cus3,mer1);
mer1 : process <Merchant>(mer1,cus1,cus2,cus3);

-----------------------------------------
-- Atomic Propositions
-----------------------------------------
DEFINE DEF_Pay := (cus1.state = c5 | cus2.state = c5 | cus3.state = c5);
DEFINE DEF_Deliver := (cus1.state = c7 | cus2.state = c7 | cus3.state = c7);

-----------------------------------------
-- The definition of Customer Agent (cus1,cus2,cus3)
-----------------------------------------
MODULE Customer (arg1,arg2)
VAR state: {c3,c4,c5,c6,c7,c8,c9,c0,c1,c2};
IVAR action :
{Reject,Null,Payment,Accept,Alpha_cus,notPayment,Request,Beta_cus,Gamma_cus};
INIT (state = c0)
       TRANS(next(state)= case
              (arg1.state = c4 & arg1.action = Null) : c0;
              (arg1.state = c5 & arg2.action = notDeliver) : c6;
              (arg1.state = c5 & arg1.action = Gamma_cus) : c3;
              (arg1.state = c5 & arg2.action = Deliver) : c7;
              (arg1.state = c5 & arg1.action = Beta_cus) : c5;
              (arg1.state = c6 & arg2.action = Refund) : c8;
              (arg1.state = c7 & arg2.action = Receipt) : c9;
              (arg1.state = c8 & arg1.action = Null) : c0;
              (arg1.state = c9 & arg1.action = Null) : c0;
              (arg1.state = c0 & arg1.action = Request) : c1;
              (arg1.state = c1 & arg2.action = Quote) : c2;
              (arg1.state = c2 & arg1.action = Reject) : c4;
              (arg1.state = c2 & arg1.action = Accept) : c3;
              (arg1.state = c3 & arg1.action = Alpha_cus) : c5;
              (arg1.state = c3 & arg1.action = Payment) : c5;
              (arg1.state = c3 & arg1.action = notPayment) : c4;
              (arg1.state = c3 & arg1.action = Beta_cus) : c3;
       esac)
-----------------------------------------
-- The definition of Merchant Agent (mer1)
-----------------------------------------
MODULE Merchant (arg1,arg2,arg3,arg4)
VAR state: {m0,m1,m2,m3,m4,m5,m6,m7,m8,m9};
IVAR action :
{Quote,Deliver,Null,Beta_mer,Receipt,Refund,notDeliver,Gamma_mer,Alpha_mer};
INIT (state = m0)
       TRANS(next(state)= case
              (arg1.state = m0 & (  arg3.action = Request | arg4.action = Request |
arg2.action = Request )) : m1;
              (arg1.state = m1 & arg1.action = Quote) : m2;
              (arg1.state = m2 & (  arg3.action = Reject | arg2.action = Reject |
arg4.action = Reject )) : m4;
              (arg1.state = m2 & (  arg3.action = Accept | arg4.action = Accept |
arg2.action = Accept )) : m3;
              (arg1.state = m3 & (  arg4.action = notPayment | arg3.action =
notPayment | arg2.action = notPayment )) : m4;
              (arg1.state = m3 & (  arg3.action = Payment | arg4.action = Payment |
arg2.action = Payment )) : m5;
              (arg1.state = m4 & arg1.action = Null) : m0;
              (arg1.state = m5 & arg1.action = Deliver) : m7;
              (arg1.state = m5 & arg1.action = notDeliver) : m6;
              (arg1.state = m5 & arg1.action = Alpha_mer) : m7;
              (arg1.state = m5 & arg1.action = Beta_mer) : m5;
              (arg1.state = m6 & arg1.action = Refund) : m8;
```

```
             (arg1.state = m7 & arg1.action = Beta_mer) : m7;
             (arg1.state = m7 & arg1.action = Gamma_mer) : m5;
             (arg1.state = m7 & arg1.action = Receipt) : m9;
             (arg1.state = m8 & arg1.action = Null) : m0;
             (arg1.state = m9 & arg1.action = Null) : m0;
      esac)

SPEC AG!((EAX(Cus.action = Gamma_Cus)(AAX(Cus.action = Alpha_Cus)(AAX(Cus.action =
Beta_Cus)(Def_Pay)&AAX(Mer.action = Beta_Mer)(Def_Pay))||EAX(Cus.action =
Beta_Cus)(EAX(Cus.action = Gamma_Cus)(AAX(Cus.action = Alpha_Cus)(AAX(Cus.action =
Beta_Cus)(Def_Pay)&AAX(Mer.action = Beta_Mer)(Def_Pay))))||EAX(Mer.action =
Beta_Mer)(EAX(Cus.action = Gamma_Cus)(AAX(Cus.action = Alpha_Cus)(AAX(Cus.action =
Beta_Cus)(Def_Pay)&AAX(Mer.action = Beta_Mer)(Def_Pay))))))&AG(! AAX(Mer.action =
Beta_Mer)(Def_Pay)))
SPEC AG!((EAX(Cus.action = Gamma_Cus)(AAX(Cus.action = Alpha_Cus)(AAX(Cus.action =
Beta_Cus)(Def_Pay)&AAX(Mer.action = Beta_Mer)(Def_Pay))||EAX(Cus.action =
Beta_Cus)(EAX(Cus.action = Gamma_Cus)(AAX(Cus.action = Alpha_Cus)(AAX(Cus.action =
Beta_Cus)(Def_Pay)&AAX(Mer.action = Beta_Mer)(Def_Pay))))||EAX(Mer.action =
Beta_Mer)(EAX(Cus.action = Gamma_Cus)(AAX(Cus.action = Alpha_Cus)(AAX(Cus.action =
Beta_Cus)(Def_Pay)&AAX(Mer.action = Beta_Mer)(Def_Pay))))))&AG(! AAX(Cus.action =
Beta_Cus)(Def_Pay)))
SPEC AG!(( AAX(Cus.action = Alpha_Cus)(AAX(Mer.action =
Beta_Mer)(Def_Pay)&AAX(Cus.action = Beta_Cus)(Def_Pay)))&AG(! AAX(Cus.action =
Beta_Cus)( AAX(Cus.action = Alpha_Cus)(AAX(Mer.action =
Beta_Mer)(Def_Pay)&AAX(Cus.action = Beta_Cus)(Def_Pay))))))
SPEC AG( AAX(Mer.action = Alpha_Mer)(AAX(Cus.action =
Beta_Cus)(Def_Deliver)&AAX(Mer.action = Beta_Mer)(Def_Deliver))->EF(EAX(Mer.action =
Gamma_Mer)(AAX(Mer.action = Alpha_Mer)(AAX(Mer.action =
Beta_Mer)(Def_Deliver)&AAX(Cus.action = Beta_Cus)(Def_Deliver))||EAX(Mer.action =
Beta_Mer)(EAX(Mer.action = Gamma_Mer)(AAX(Mer.action = Alpha_Mer)(AAX(Mer.action =
Beta_Mer)(Def_Deliver)&AAX(Cus.action = Beta_Cus)(Def_Deliver))))||EAX(Cus.action =
Beta_Cus)(EAX(Mer.action = Gamma_Mer)(AAX(Mer.action = Alpha_Mer)(AAX(Mer.action =
Beta_Mer)(Def_Deliver)&AAX(Cus.action = Beta_Cus)(Def_Deliver)))))))
SPEC EF AAX(Mer.action = Alpha_Mer)(AAX(Cus.action =
Beta_Cus)(Def_Deliver)&AAX(Mer.action = Beta_Mer)(Def_Deliver))
SPEC AG!( AAX(Mer.action = Alpha_Mer)(AAX(Cus.action =
Beta_Cus)(Def_Deliver)&AAX(Mer.action = Beta_Mer)(Def_Deliver)))

***Java Tranformer Tool Finished.***
```