MATES 2010-Eighth German Conference on Multi Agents System Technologies

September 27 - 29, 2010 in Leipzig, Germany

A Novel Formal Specification Approach for Real Time Multi-Agent System Functional Requirements

By: Mohamed Amin LAOUADI

Farid MOKHATI

Hassina SERIDI-Bouchelaghem

Plan

- Introduction
- Related Works
- Real Time Maude
- Extended Agent UML
- The Proposed Approach
- Case Study: Supply Chain Management (SCM)
- Conclusion and Future Work

Introduction

- The real time MAS reflect intrinsic real-time systems characteristics, more precisely, the time constraints.
- MAS designers have development methodologies and modeling language.
- None of the proposed methodologies takes into account the functional requirements formalization.
- A use case oriented specification of real time MAS functional requirements.

Introduction (2)

- AUML models suffer as UML of a lack of formal semantics.
- Formal methods represent an interesting solution.
- The main interest in this work is to describe :
  - the functional requirements of real time MAS using Agent UML
  - Translate these semi-formal descriptions in RT-Maude.

Related Works

- Among the methodologies that directly addressing the design of real-time multi-agent systems:
  - RT-Message,
  - the extended BDI-ASDP methodology for real time,
  - the development method of Lichen Zhang.
- For a description of real-time agents : domain model, role model, and timed model.

Related Works (2)

- These methodologies don't focus on the real time MAS functional requirements formalization.
- they will supplemented by methods that strongly encourage the formalization of the functional requirements.

Real Time Maude

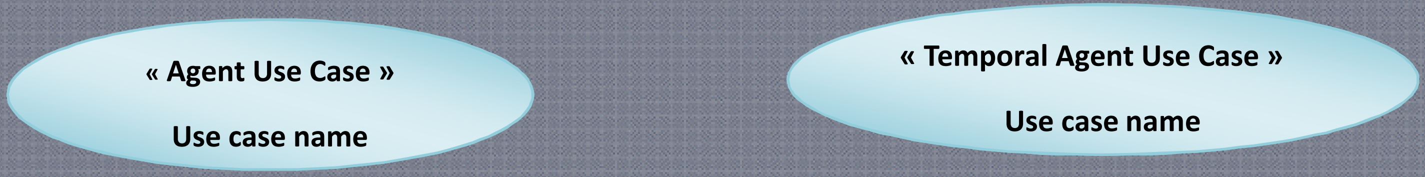
- RT-Maude contains the specification of :
  - ○ sort Time to describe the time domain,,
  - ○ sort GlobalSystem with a constructor '{_}'': {_} : System -> GlobalSystem
  - ○ a set of tick rules : {t} => {t'} in time u if cond .
- Real-time rewrite theories are specified in RT-Maude as :

```
(tmod    NAME  is   ...     endtm)
```

```
(tomod    NAME  is   ...     endtom)
```

Extended Agent UML

- Adapting AUML to the description of RT-MAS functional requirements.
- Use case diagrams will be enriched by the following 'five stereotypes':



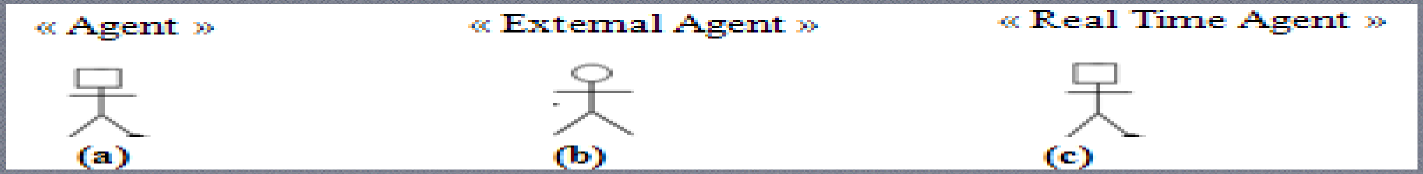
« Agent Use Case »

Use case name

« Temporal Agent Use Case »

Use case name

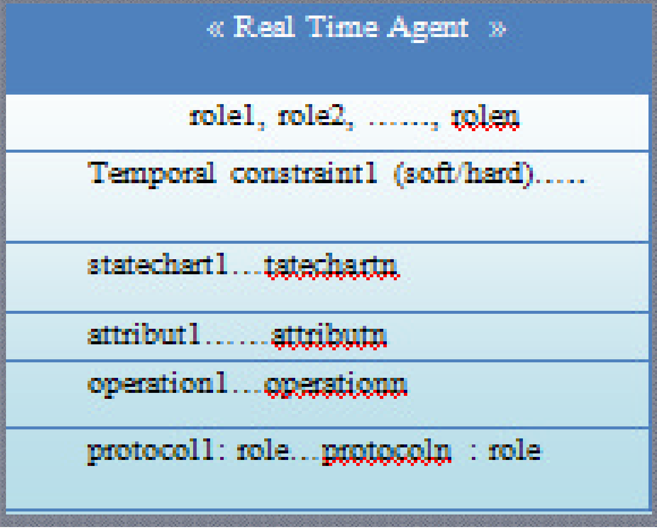
Stereotyped use cases



Agent notations

Extended Agent UML (2)

- To all compartments proposed in Huget class diagrams, a new compartment called "temporal constraints" is added:



« Real Time Agent »

role1, role2, ……, rolen

Temporal constraint1 (soft/hard)…..

statechart1…tatechartn

attribut1……attributn

operation1…operationn

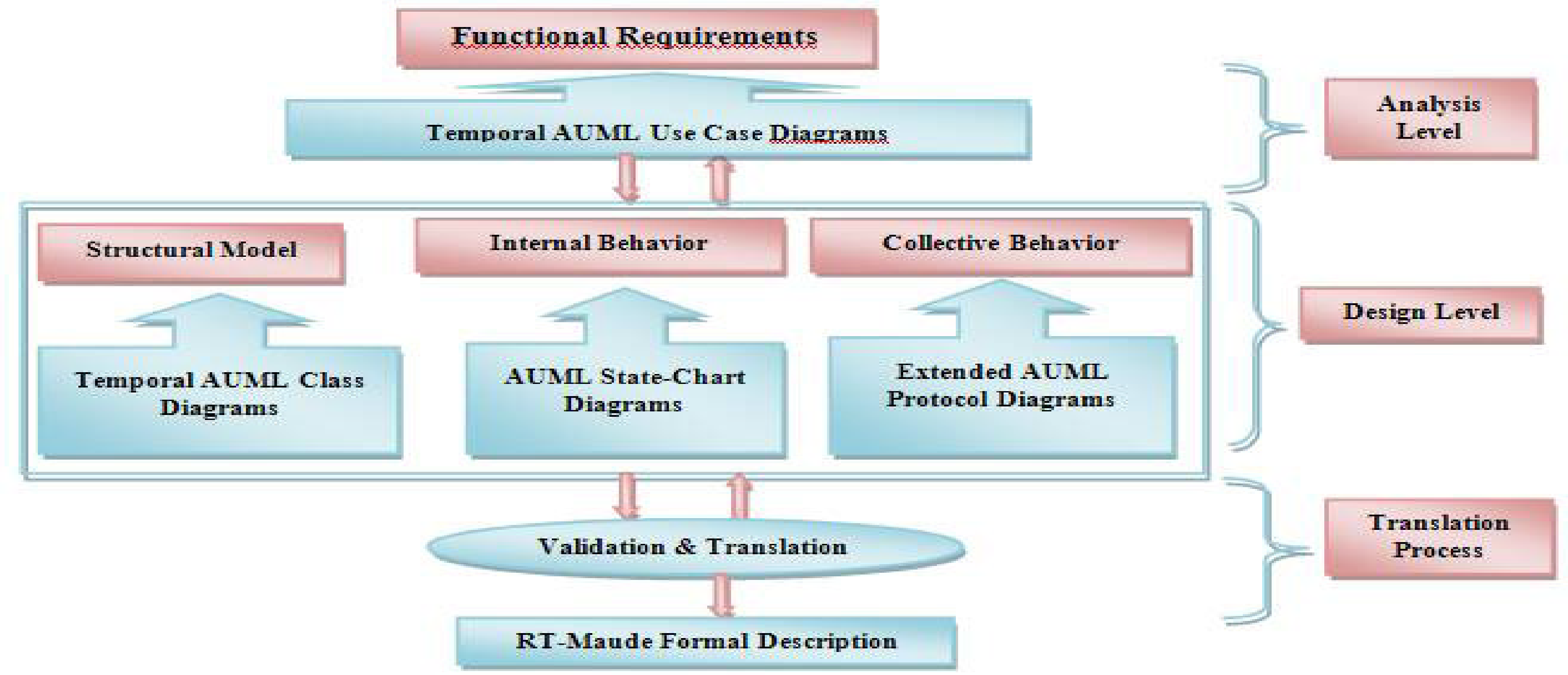
protocol1: role…protocoln : role

The used AUML class diagram

- To describe agents' individual and collective behaviors we use respectively the AUML state-chart and protocol diagrams.

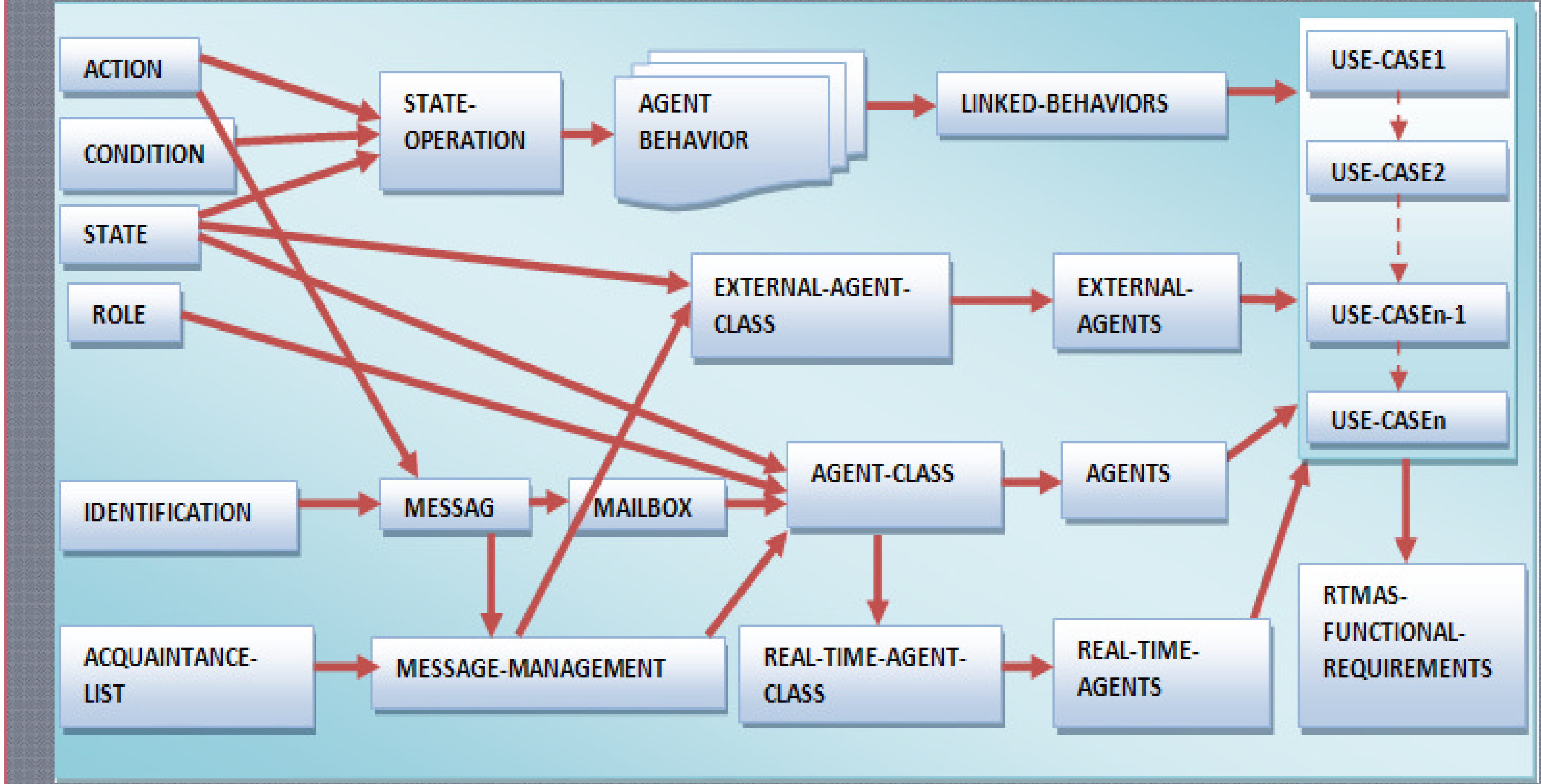
The Proposed Approach

- The translation process is divided into three major steps :



Methodology of the approach

The Proposed Approach(2)



Generated Modules

The Proposed Approach (3)

```
 (omod EXTERNAL-AGENT-CLASS is
protecting STATE .      protecting MESSAGE-MANAGEMENT .
class ExtAgent | CurrentState : State, AcqList :
                 AcquaintanceList   .
endom)
```

The O.O Module EXTERNAL-AGENT-CLASS

```
 (omod AGENT-CLASS is
protecting STATE .   protecting ROLE .
protecting MAILBOX .    protecting MESSAGE-MANAGEMENT .
class Agent | CurrentState : State, PlayRole : AgentRole,
              AcqList : AcquaintanceList, MBox : MailBox .
endom)
```

The O.O Module AGENT-CLASS

The Proposed Approach (4)

```
(tomod REAL-TIME-AGENT-CLASS is
extending AGENT-CLASS   .
class RealTimeAgent | Clock : Time .
subclass RealTimeAgent  <  Agent .     endtom)
```

The Timed O.O Module REAL-TIME-AGENT-CLASS

```
(tomod USE-CASEi is
********* User  Part
including NAT-TIME-DOMAIN .   including EXTERNAL-AGENTS.
including AGENTS .  including REAL-TIME-AGENTS .
including  LINKED-BEHAVIORS . subsort String < Identifier .
rl [1] : Configuration1 => Configuration2.
…
rl [m] : Configuration 2m-1 => <Configuration2m .    endtom)
```

The Timed O.O Module USE-CASEi

The Proposed Approach (5)

```
(tomod RTMAS-FUNCTIONAL-REQUIREMENTS  is
including USE-CASE1 .
including USE-CASE2 .
****…
including USE-CASEm .  endtom)
```

The Timed O.O Module RTMAS-FUNCTIONAL-REQUIREMENTS

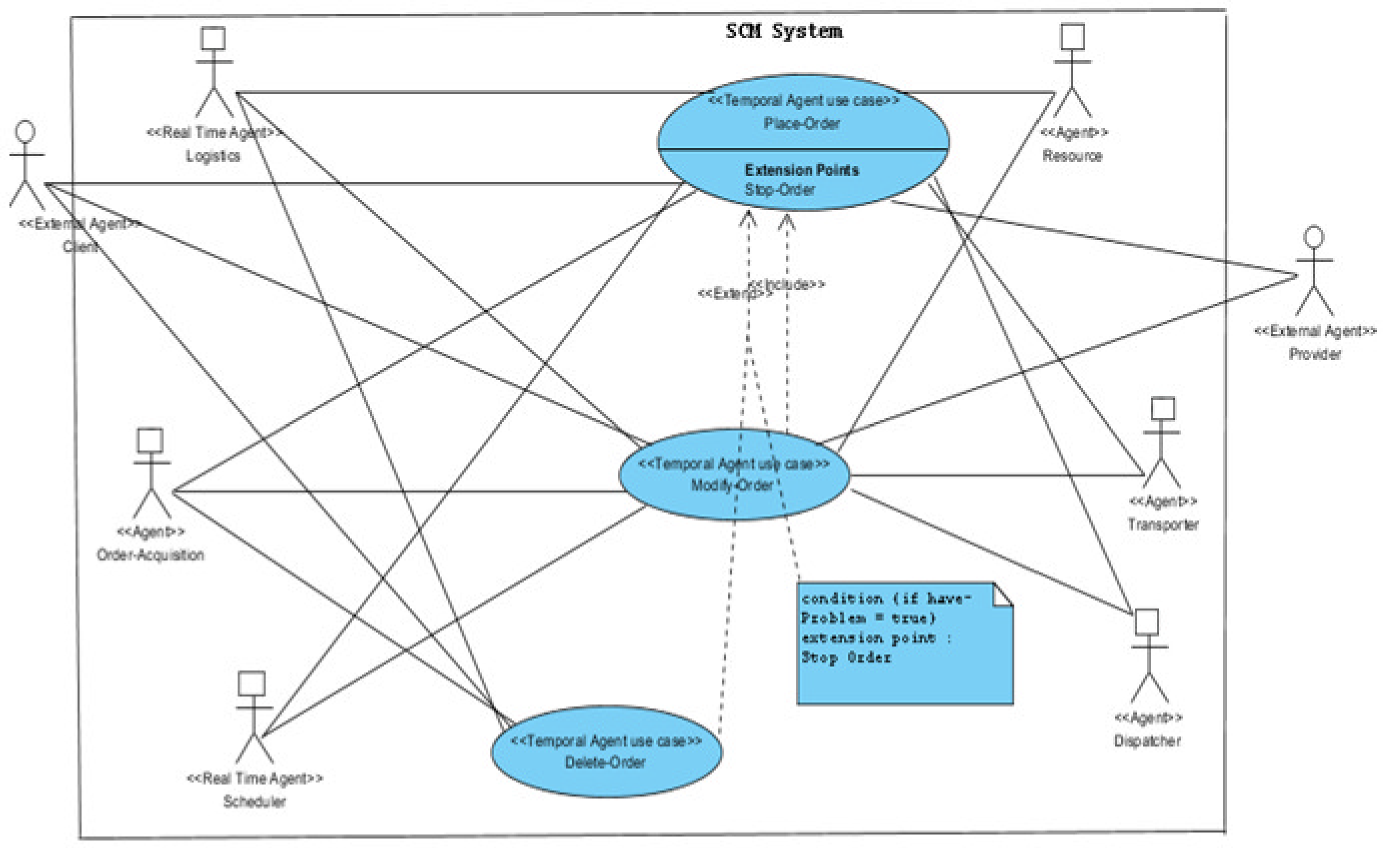
```
crl [tick] :  {Timer(TimeOut)
< A : RealTimeAgent | Clock : T, CurrentState : S >
REST:Configuration}
 =>  {Timer(TimeOut monus 1)
 < A : RealTimeAgent | Clock : T plus 1, CurrentState : S >
 REST:Configuration} in time 1
 if  (TimeOut > zero)  and (S == AgentState (WaitL)) .
```

The Tick Rule

Case Study: Supply Chain Management (SCM)

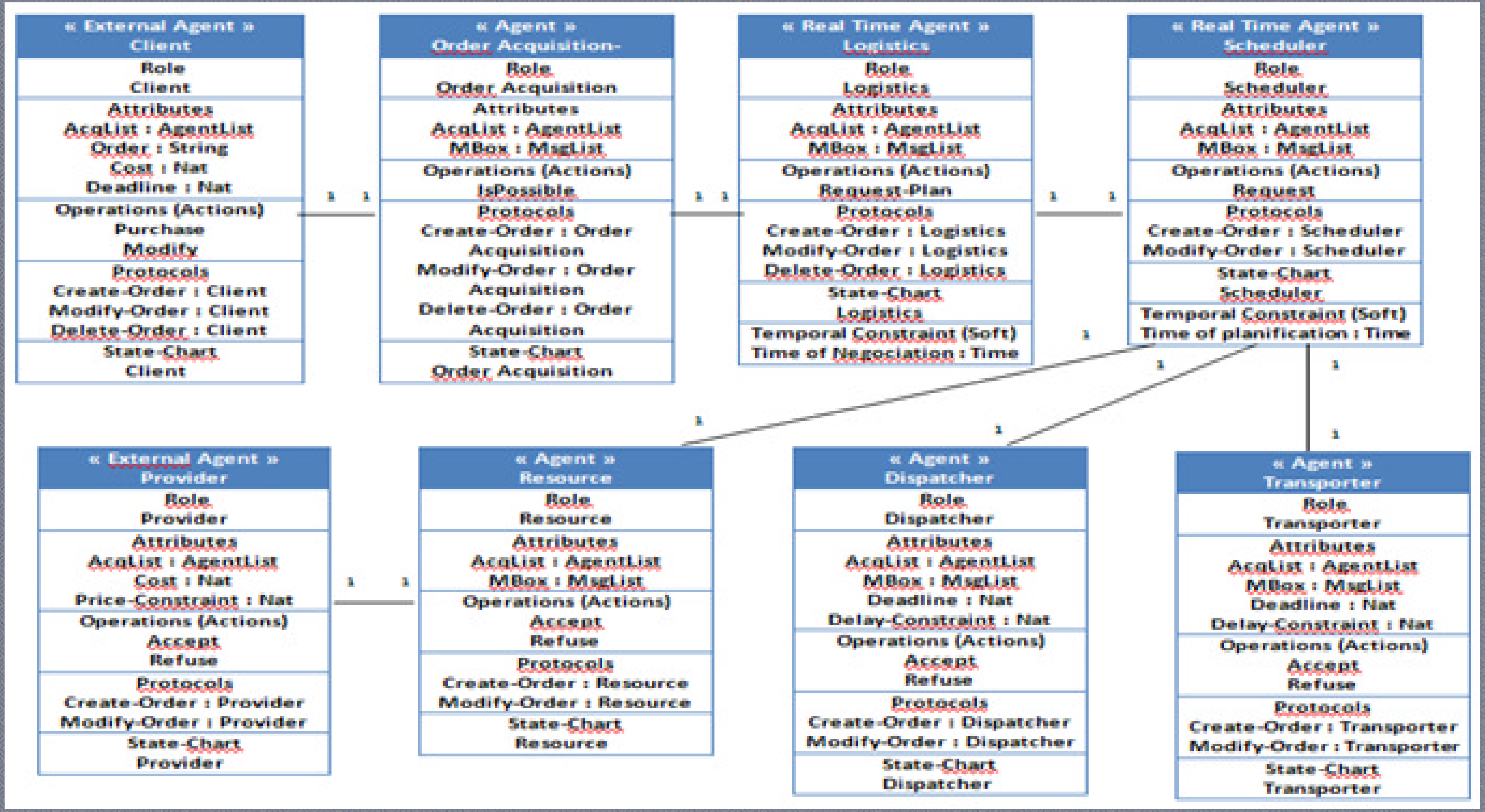
- Supply Chain Management Modeling.
- Translation Process Application.
- Generated Description Validation.

Supply Chain Management Modeling



AUML Use Case Diagram of SCM

Supply Chain Management Modeling (2)



« External Agent »
Client
Role
Client
Attributes
AcqList : AgentList
Order : String
Cost : Nat
Deadline : Nat
Operations (Actions)
Purchase
Modify
Protocols
Create-Order : Client
Modify-Order : Client
Delete-Order : Client
State-Chart
Client

« Agent »
Order Acquisition-
Role
Order Acquisition
Attributes
AcqList : AgentList
MBox : MsgList
Operations (Actions)
IsPossible
Protocols
Create-Order : Order Acquisition
Modify-Order : Order Acquisition
Delete-Order : Order Acquisition
State-Chart
Order Acquisition

« Real Time Agent »
Logistics
Role
Logistics
Attributes
AcqList : AgentList
MBox : MsgList
Operations (Actions)
Request-Plan
Protocols
Create-Order : Logistics
Modify-Order : Logistics
Delete-Order : Logistics
State-Chart
Logistics
Temporal Constraint (Soft)
Time of Negociation : Time

« Real Time Agent »
Scheduler
Role
Scheduler
Attributes
AcqList : AgentList
MBox : MsgList
Operations (Actions)
Request
Protocols
Create-Order : Scheduler
Modify-Order : Scheduler
State-Chart
Scheduler
Temporal Constraint (Soft)
Time of planification : Time

« External Agent »
Provider
Role
Provider
Attributes
AcqList : AgentList
Cost : Nat
Price-Constraint : Nat
Operations (Actions)
Accept
Refuse
Protocols
Create-Order : Provider
Modify-Order : Provider
State-Chart
Provider

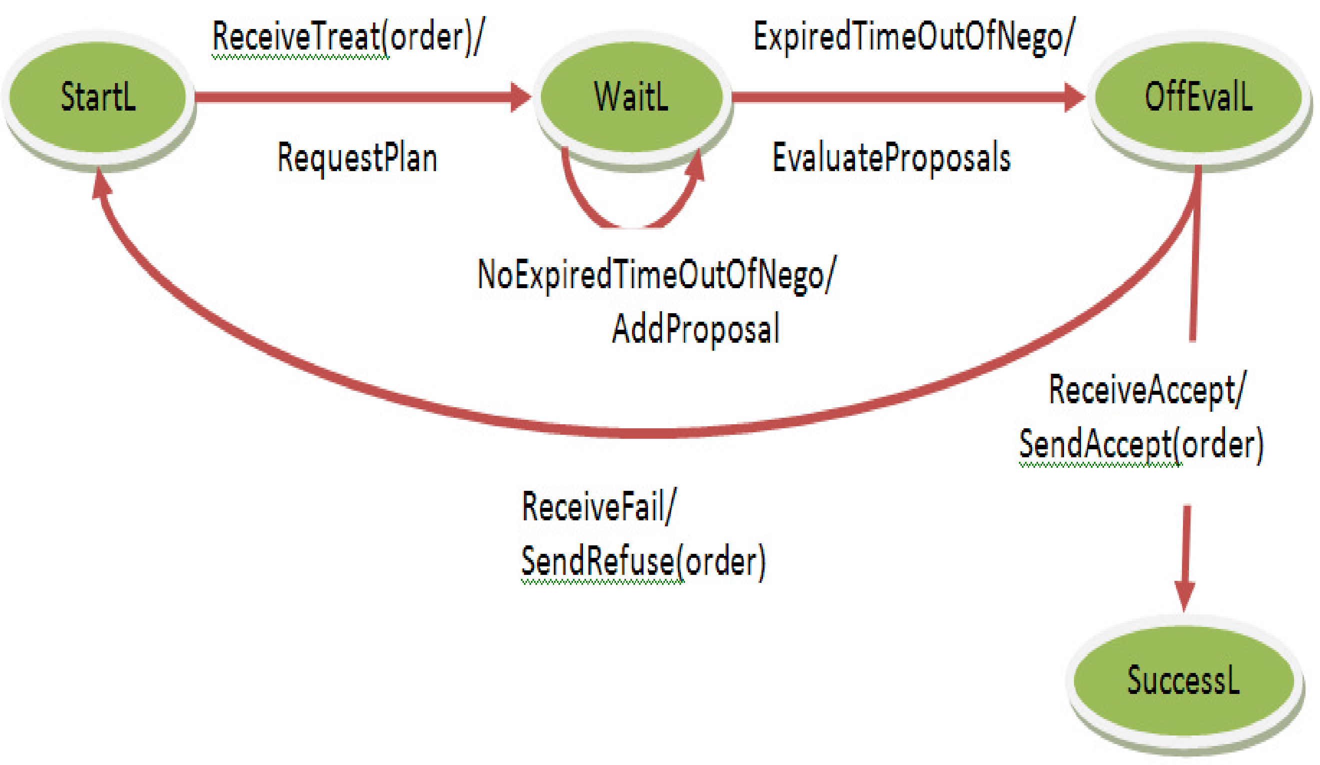
« Agent »
Resource
Role
Resource
Attributes
AcqList : AgentList
MBox : MsgList
Operations (Actions)
Accept
Refuse
Protocols
Create-Order : Resource
Modify-Order : Resource
State-Chart
Resource

« Agent »
Dispatcher
Role
Dispatcher
Attributes
AcqList : AgentList
MBox : MsgList
Deadline : Nat
Delay-Constraint : Nat
Operations (Actions)
Accept
Refuse
Protocols
Create-Order : Dispatcher
Modify-Order : Dispatcher
State-Chart
Dispatcher

« Agent »
Transporter
Role
Transporter
Attributes
AcqList : AgentList
MBox : MsgList
Deadline : Nat
Delay-Constraint : Nat
Operations (Actions)
Accept
Refuse
Protocols
Create-Order : Transporter
Modify-Order : Transporter
State-Chart
Transporter

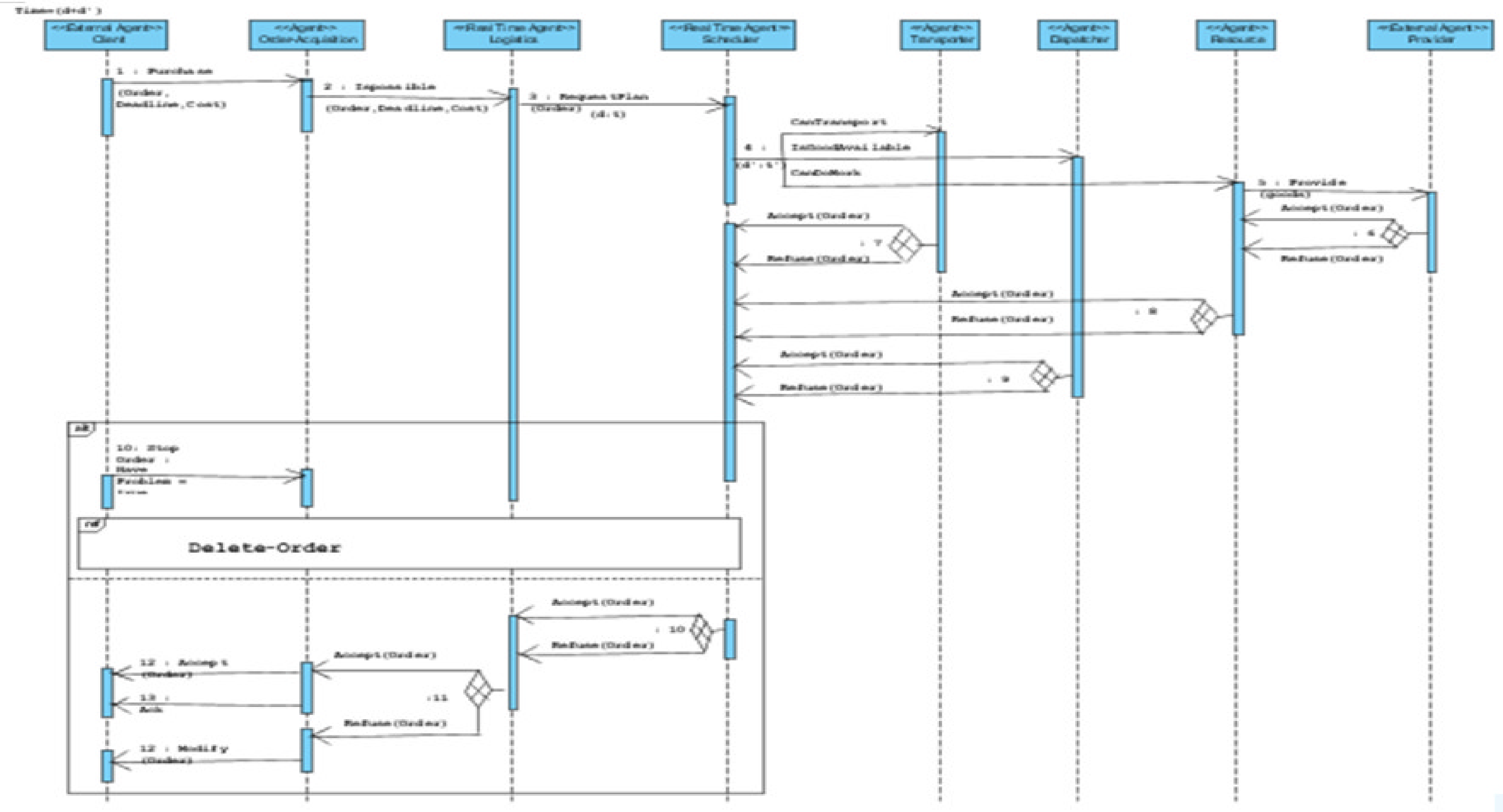
AUML class Diagram of SCM

Supply Chain Management Modeling (3)



AUML State-chart diagram of the Real Time Agent Logistics

Supply Chain Management Modeling (4)



AUML Protocol diagram corresponding to use case Place-Order

Translation Process Application

```
(omod EXTERNAL-AGENTS is
extending EXTERNAL-AGENT-CLASS. including STRING.
including NAT.
subclass Client <  ExtAgent . subclass Provider <  ExtAgent .
class Client | Order : String, Deadline : Nat, Cost : Nat  .
class Provider | PriceConstraint : Nat, Cost : Nat  .
endom)
```

The O.O Module EXTERNAL-AGENTS

```
(omod AGENTS is extending AGENT-CLASS .
including STRING .    including NAT .
subclass Transporter < Agent . subclass Dispatcher < Agent .
class Transporter | DelayConstraint : Nat, Deadline : Nat.
class Dispatcher |DelayConstraint : Nat,Deadline : Nat.
endom)
```

The O.O Module AGENTS

Translation Process Application (2)

```
(tomod PLACE-ORDER is
including NAT-TIME-DOMAIN .  including EXTERNAL-AGENTS .
ops Event GetEvent :  Identifier State Condition -> Msg .
vars A A1 : Identifier. vars S S1 : State. vars MB : MailBox.
crl [Internal-Log2] : GetEvent(A, S, Cond)       Execute(Act)
< A : RealTimeAgent |PlayRole : Logistics, CurrentState : S >
⇒ < A : RealTimeAgent | PlayRole : Logistics, CurrentState :
  TargetState(S, Cond) >   if (Cond == ExpiredTimeOutOfNeg)
  and(IsInternalAction(ActionToAccomplish(S, Cond))== true) .
crl [Internal-Sch2] : GetEvent(A, S, Cond) Execute(Act)
< A : RealTimeAgent |PlayRole : Scheduler, CurrentState : S >
=> < A : RealTimeAgent | PlayRole : Scheduler, CurrentState :
  TargetState(S, Cond) > if (Cond == ExpiredTimeOutOfSched)and
  (IsInternalAction (ActionToAccomplish(S, Cond))  ==  true)
  ****...
```

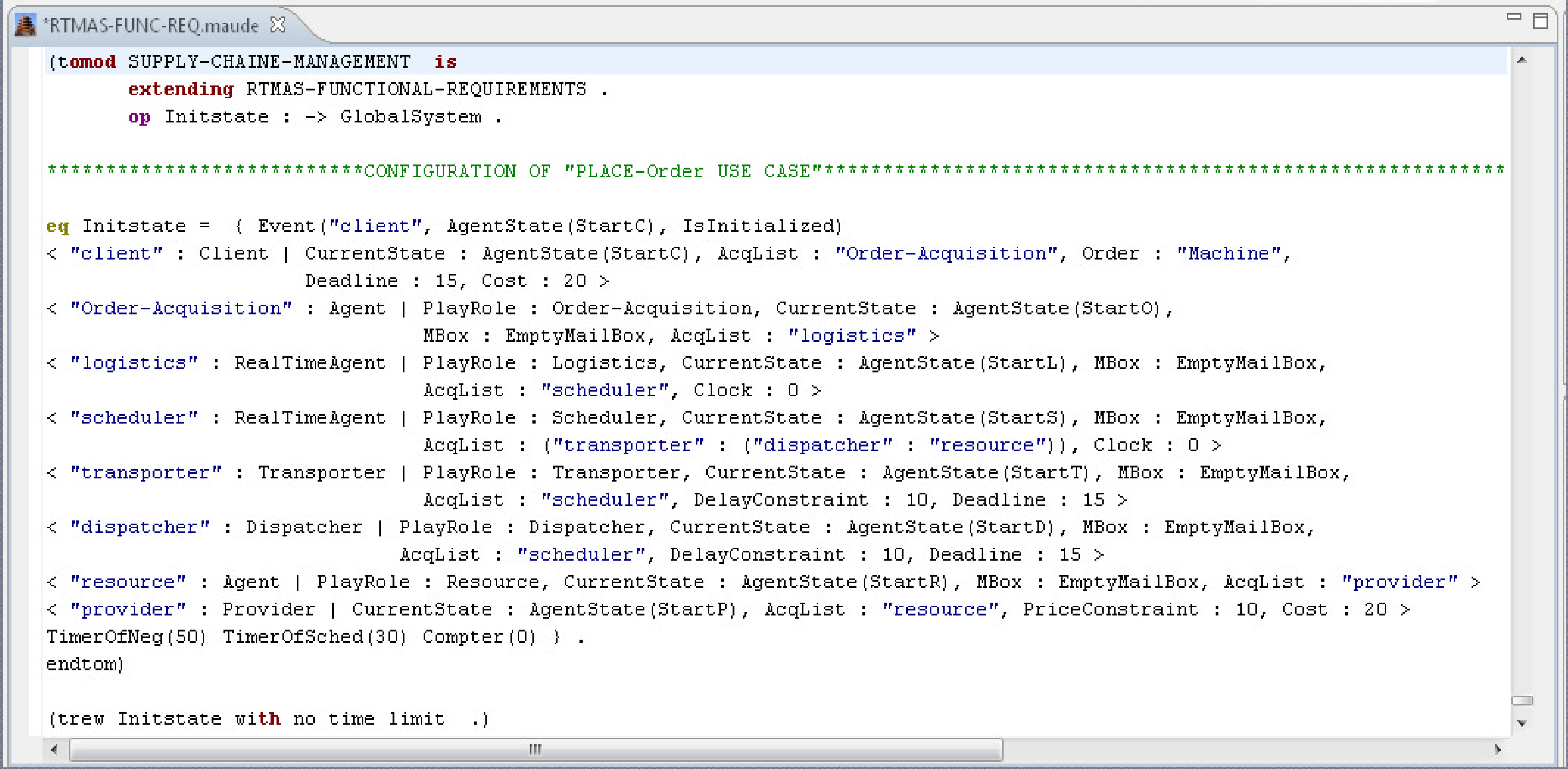
The Timed O.O Module PLACE-ORDER

Translation Process Application (3)

```
(tomod RTMAS-FUNCTIONAL-REQUIREMENTS is
including PLACE-ORDER .
including MODIFY-ORDER .
including DELETE-ORDER .
endtom)
```

The Timed O.O Module RTMAS-FUNCTIONAL-REQUIREMENTS

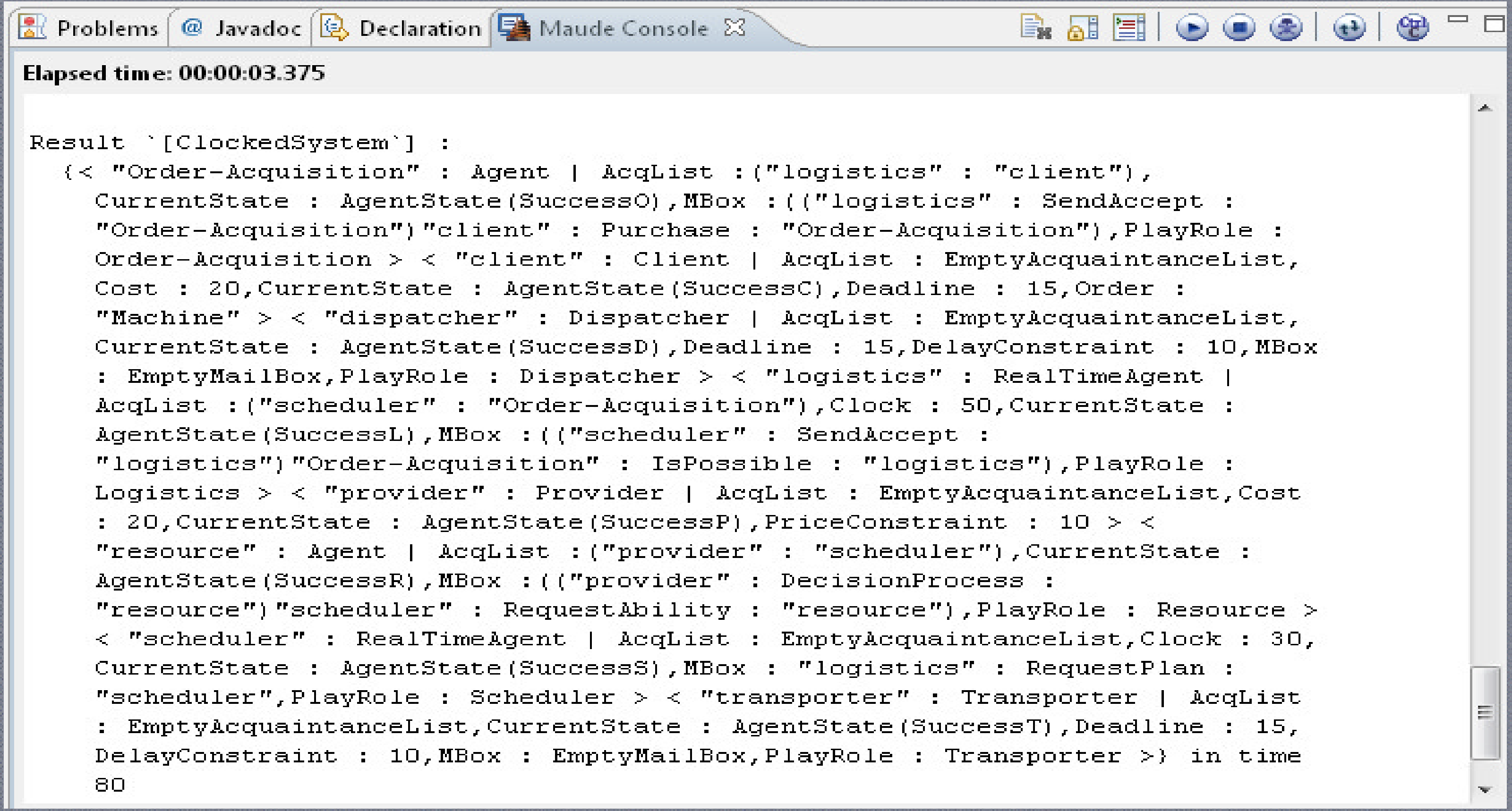
Generated Description Validation



```
(tomod SUPPLY-CHAINE-MANAGEMENT  is
       extending RTMAS-FUNCTIONAL-REQUIREMENTS .
       op Initstate : -> GlobalSystem .

***************************CONFIGURATION OF "PLACE-Order USE CASE"***********************************************************

eq Initstate =  { Event("client", AgentState(StartC), IsInitialized)
< "client" : Client | CurrentState : AgentState(StartC), AcqList : "Order-Acquisition", Order : "Machine",
                      Deadline : 15, Cost : 20 >
< "Order-Acquisition" : Agent | PlayRole : Order-Acquisition, CurrentState : AgentState(StartO),
                                MBox : EmptyMailBox, AcqList : "logistics" >
< "logistics" : RealTimeAgent | PlayRole : Logistics, CurrentState : AgentState(StartL), MBox : EmptyMailBox,
                                AcqList : "scheduler", Clock : 0 >
< "scheduler" : RealTimeAgent | PlayRole : Scheduler, CurrentState : AgentState(StartS), MBox : EmptyMailBox,
                                AcqList : ("transporter" : ("dispatcher" : "resource")), Clock : 0 >
< "transporter" : Transporter | PlayRole : Transporter, CurrentState : AgentState(StartT), MBox : EmptyMailBox,
                                AcqList : "scheduler", DelayConstraint : 10, Deadline : 15 >
< "dispatcher" : Dispatcher | PlayRole : Dispatcher, CurrentState : AgentState(StartD), MBox : EmptyMailBox,
                              AcqList : "scheduler", DelayConstraint : 10, Deadline : 15 >
< "resource" : Agent | PlayRole : Resource, CurrentState : AgentState(StartR), MBox : EmptyMailBox, AcqList : "provider" >
< "provider" : Provider | CurrentState : AgentState(StartP), AcqList : "resource", PriceConstraint : 10, Cost : 20 >
TimerOfNeg(50) TimerOfSched(30) Compter(0) } .
endtom)

(trew Initstate with no time limit  .)
```

Initial Configuration

Generated Description Validation (2)



Elapsed time: 00:00:03.375

```
Result `[ClockedSystem`] :
  {< "Order-Acquisition" : Agent | AcqList :("logistics" : "client"),
    CurrentState : AgentState(SuccessO),MBox :(("logistics" : SendAccept :
    "Order-Acquisition")"client" : Purchase : "Order-Acquisition"),PlayRole :
    Order-Acquisition > < "client" : Client | AcqList : EmptyAcquaintanceList,
    Cost : 20,CurrentState : AgentState(SuccessC),Deadline : 15,Order :
    "Machine" > < "dispatcher" : Dispatcher | AcqList : EmptyAcquaintanceList,
    CurrentState : AgentState(SuccessD),Deadline : 15,DelayConstraint : 10,MBox
    : EmptyMailBox,PlayRole : Dispatcher > < "logistics" : RealTimeAgent |
    AcqList :("scheduler" : "Order-Acquisition"),Clock : 50,CurrentState :
    AgentState(SuccessL),MBox :(("scheduler" : SendAccept :
    "logistics")"Order-Acquisition" : IsPossible : "logistics"),PlayRole :
    Logistics > < "provider" : Provider | AcqList : EmptyAcquaintanceList,Cost
    : 20,CurrentState : AgentState(SuccessP),PriceConstraint : 10 > <
    "resource" : Agent | AcqList :("provider" : "scheduler"),CurrentState :
    AgentState(SuccessR),MBox :(("provider" : DecisionProcess :
    "resource")"scheduler" : RequestAbility : "resource"),PlayRole : Resource >
    < "scheduler" : RealTimeAgent | AcqList : EmptyAcquaintanceList,Clock : 30,
    CurrentState : AgentState(SuccessS),MBox : "logistics" : RequestPlan :
    "scheduler",PlayRole : Scheduler > < "transporter" : Transporter | AcqList
    : EmptyAcquaintanceList,CurrentState : AgentState(SuccessT),Deadline : 15,
    DelayConstraint : 10,MBox : EmptyMailBox,PlayRole : Transporter >} in time
    80
```

Result of the unlimited rewriting of the initial Configuration

Conclusion and Future Work

- The proposed approach considers jointly functional, static and dynamic aspects of real-time MAS.
- Using formal notations to specify RT-MAS' requirements makes it possible to produce precise descriptions.
- The approach is generic supporting the formal description and validation of RT-MAS functional requirements.

Conclusion and Future Work (2)

- Extending our approach by integrating possibilities offered by RT-Maude to verify some properties of the specification of RT-MAS' functional requirements.

- Development of a tool supporting our approach.

ThANK YOU