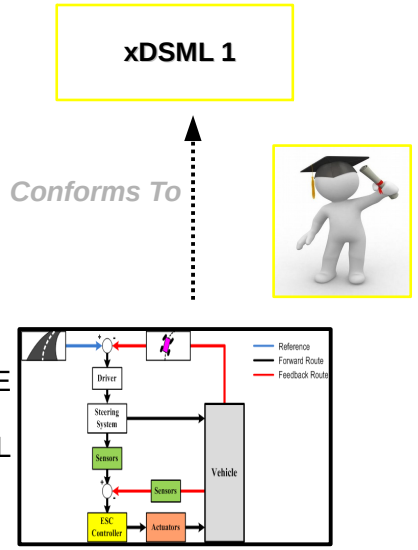
# The *B*ehavioral *C*oordination *O*perat*o*r *L*anguage

***Gemoc Final Workshop, March 17th, 2016***

Julien Deantoni
University of Nice, I3S CNRS, INRIA AOSTE
Julien.deantoni@polytech.unice.fr

# Discrete Time Behavior

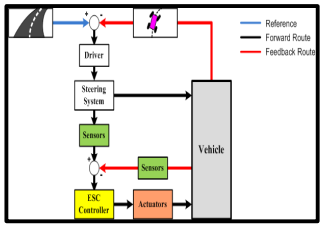



# Discrete Time Behavior

xDSML 1

Conforms To

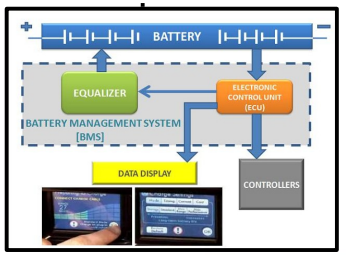
ADAPTIVE CRUISE CONTROL SYSTEM




xDSML 2

Conforms To

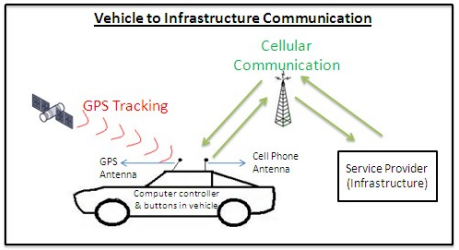
BATTERY CONTROL




xDSML 3

Conforms To

GPS TRACKING




Heterogeneous models and languages

Emerging System Behavior difficult to apprehend

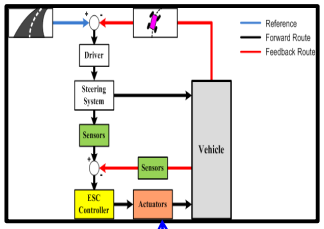
Gemoc

# Discrete Time Behavior

xDSML 1

Conforms To

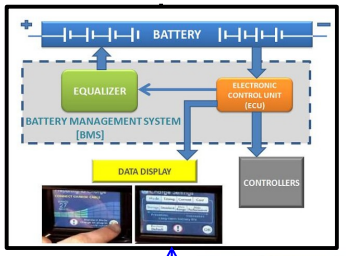
ADAPTIVE CRUISE CONTROL SYSTEM




xDSML 2

Conforms To

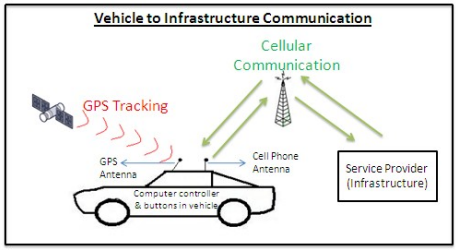
BATTERY CONTROL

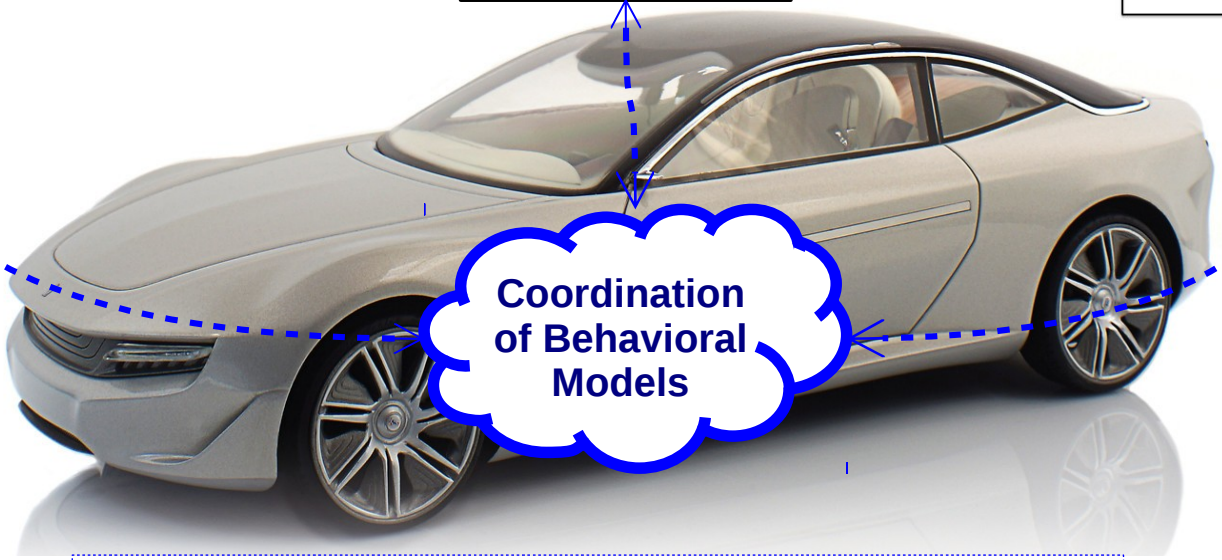



xDSML 3

Conforms To

GPS TRACKING



Know-how

System Designer

**Coordination of Behavioral Models**

Heterogeneous models and languages

Emerging System Behavior difficult to apprehend

Gemoc

# Outline

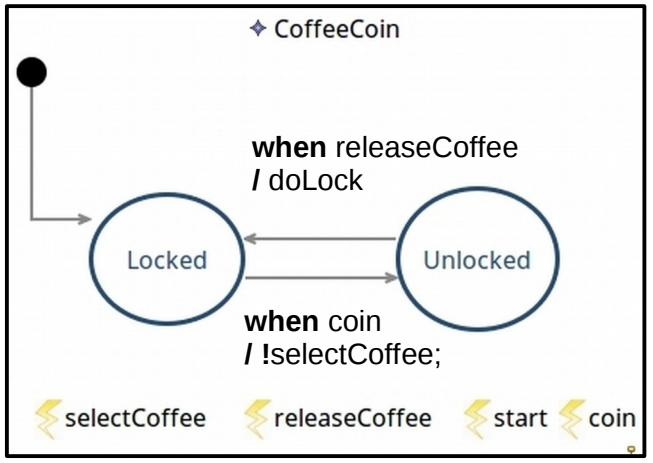
- State of the Art:
  - Coordination Languages and ADLs
  - Coordination Frameworks
- Our proposal:
  - The **B**ehavioral **C**oordination **O**perat***o***r **L**anguage
- Conclusion

# Running Example: the Coffee Machine

**Timed Finite State Machine (TFSM)**

State-based language

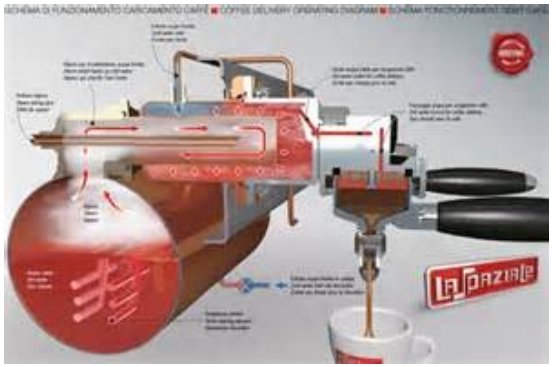
Conforms To

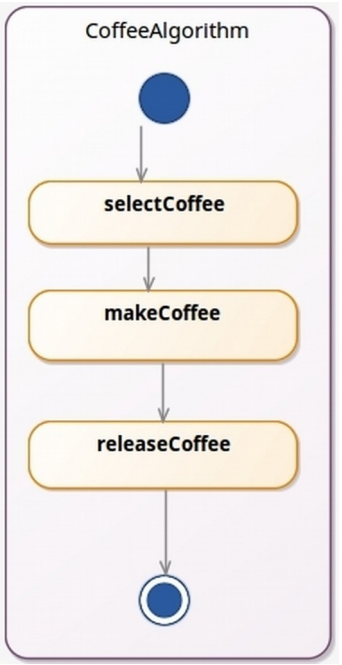



**Activity**

Action-based language

Conforms To






Gemoc

# Running Example: the Coffee Machine

**Timed Finite State Machine (TFSM)**

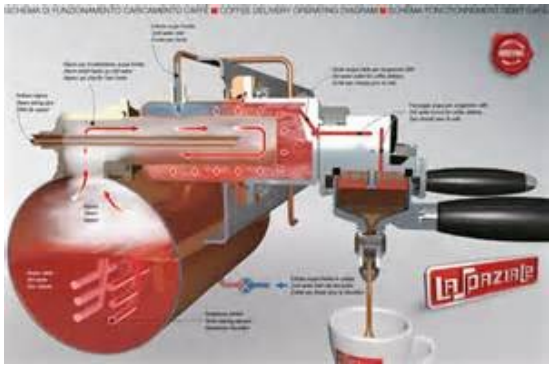
State-based language

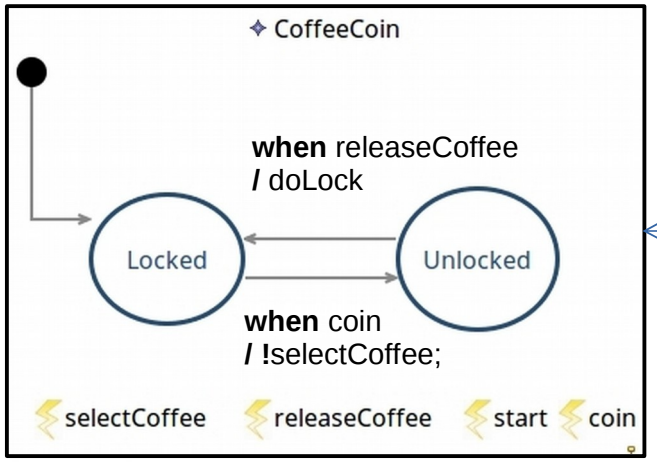
Action-based language

**Activity**

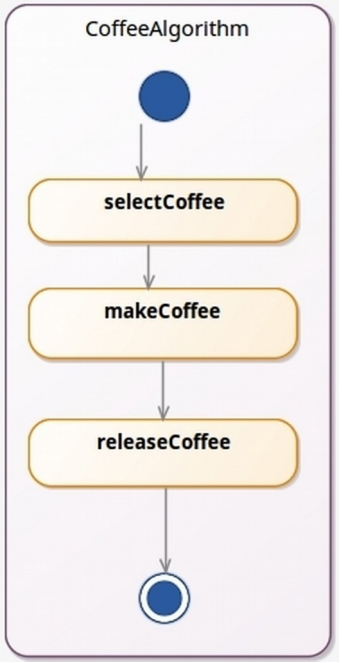
Conforms To

Conforms To






?




Gemoc

# Running Example: the Coffee Machine

**Timed Finite State Machine (TFSM)**

State-based language

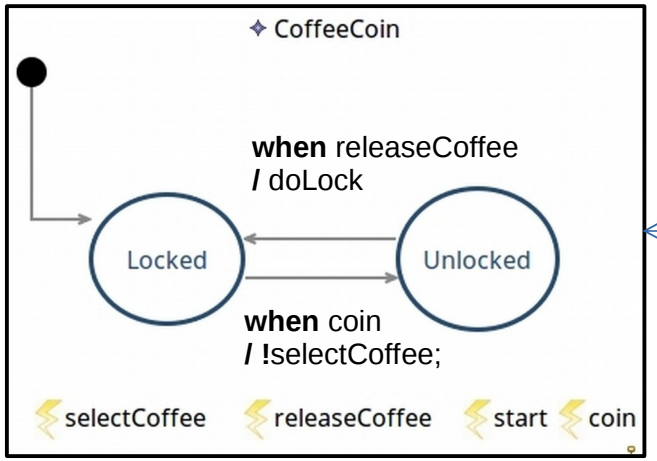
Action-based language

**Activity**

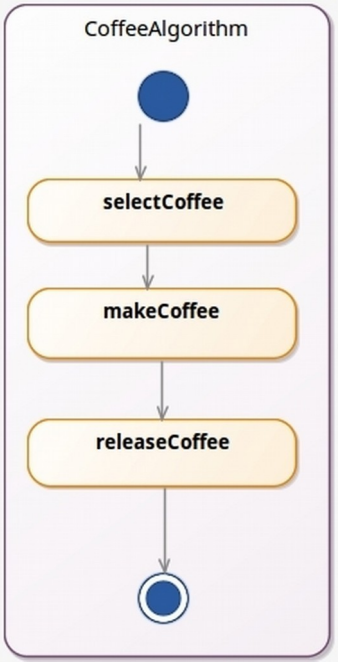
Conforms To

Conforms To

e.g.,When the event selectCoffee occurs, the Action selectCoffee is started




?




Gemoc

# Coordination Languages & ADLs

TFSM

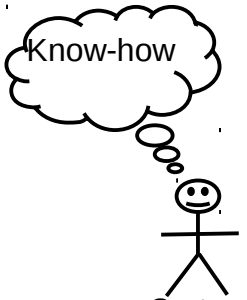
Activity

Coordination Languages proposed to model the coordination by using a dedicated Language, e.g., Linda or Esper.

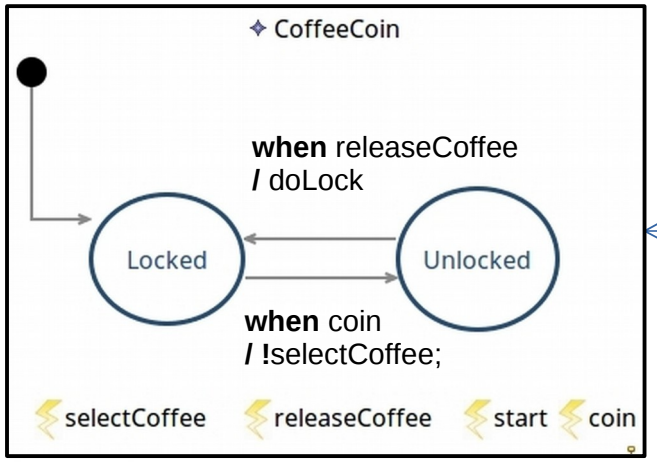
ADLs proposed component based languages to coordinate, possibly heterogeneous behaviors, e.g., Wright, Rapide or MetaH.

Conforms To

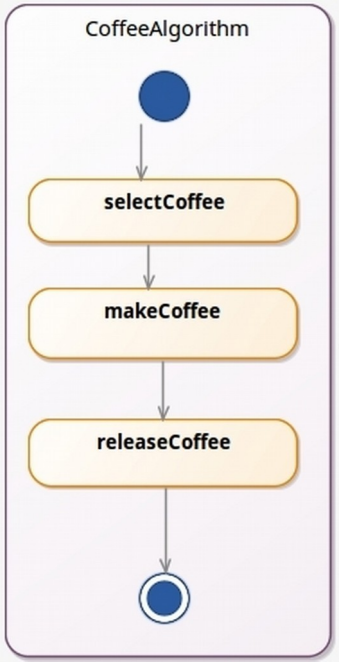
Conforms To




defines




Model of Coordination




Gemoc

# Coordination Languages & ADLs

TFSM

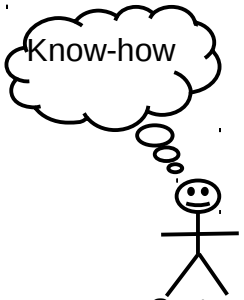
Activity

Coordination Languages proposed to model the coordination by using a dedicated Language, e.g., Linda or Esper.

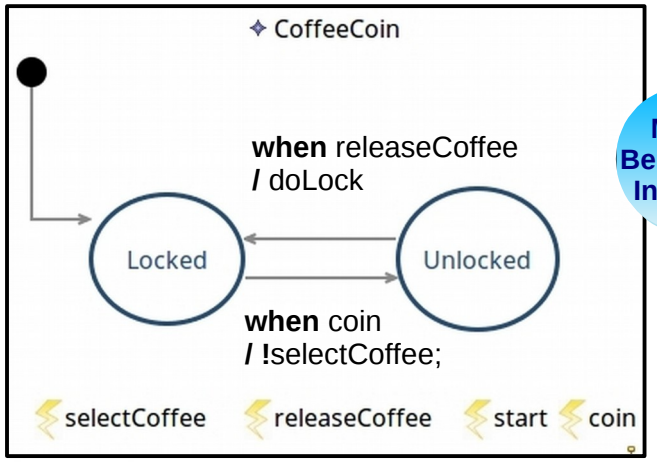
ADLs proposed component based languages to coordinate, possibly heterogeneous behaviors, e.g., Wright, Rapide or MetaH.

Conforms To

Conforms To



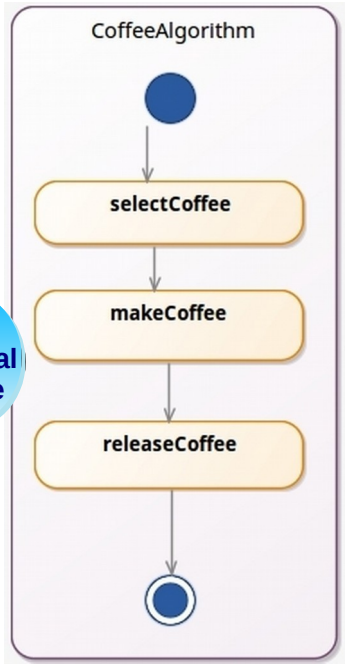

defines




Model Behavioral Interface

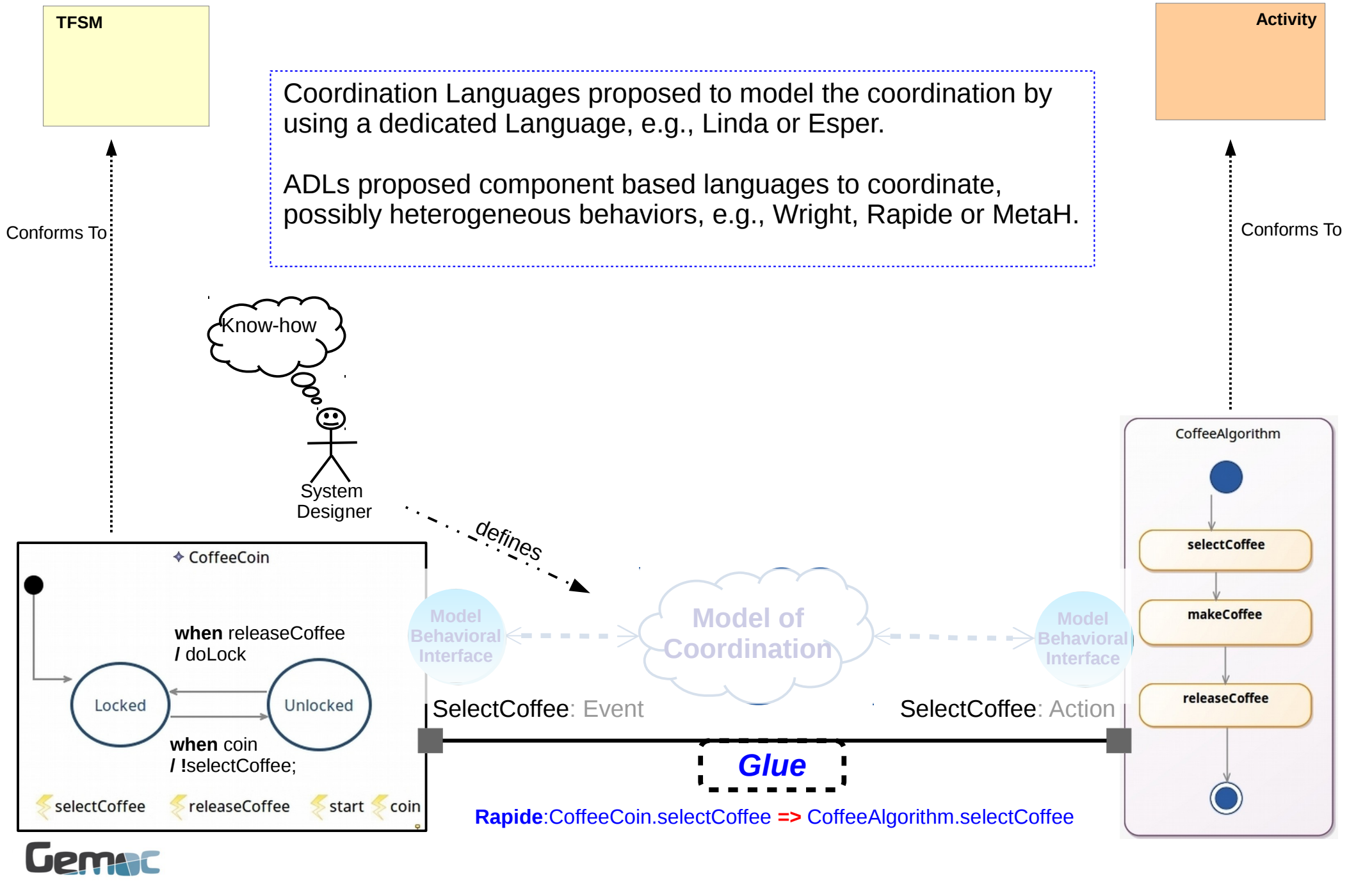
Model of Coordination

Model Behavioral Interface




Gemoc

# Coordination Languages & ADLs



# Coordination Languages & ADLs

TFSM

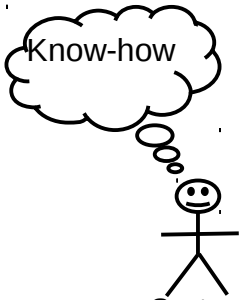
Activity

Coordination Languages proposed to model the coordination by using a dedicated Language, e.g., Linda or Esper.

ADLs proposed component based languages to coordinate, possibly heterogeneous behaviors, e.g., Wright or MetaH.

Conforms To

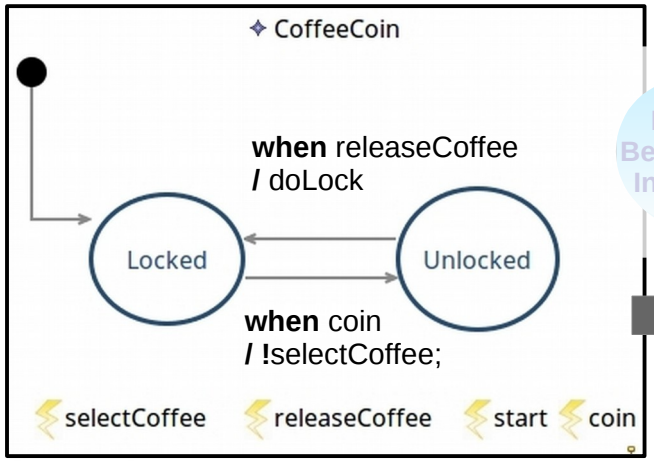
Conforms To




The coordination is modeled explicitly

The coordination is defined rule by rule

defines




Model Behavioral Interface

Model of Coordination

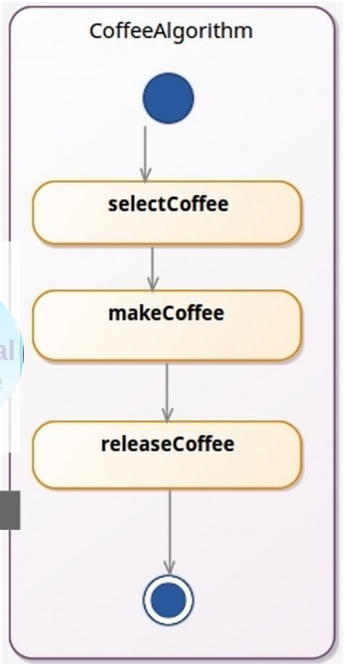
Model Behavioral Interface

SelectCoffee: Event

SelectCoffee: Action

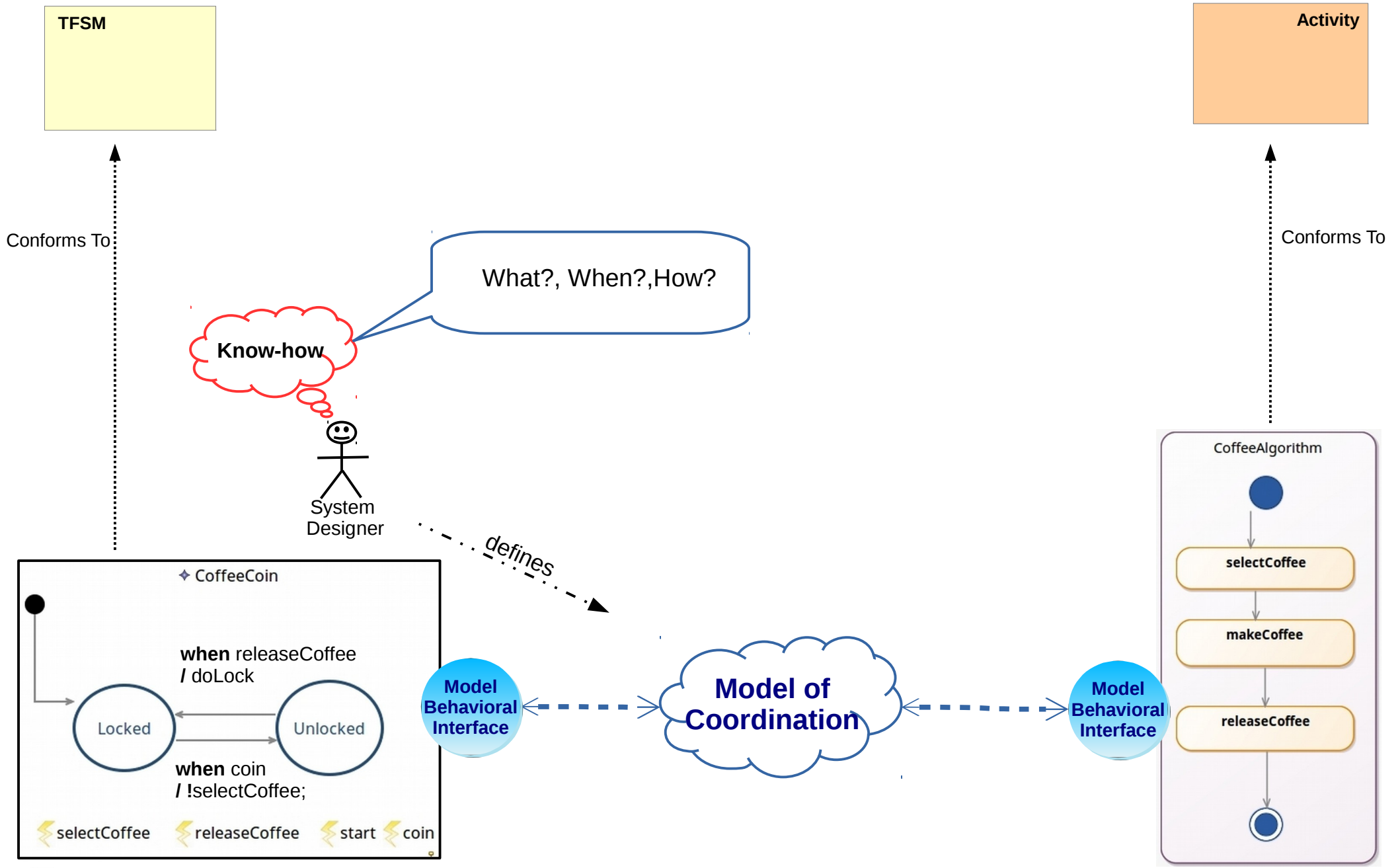
***Glue***

**Rapide**:CoffeeCoin.selectCoffee => CoffeeAlgorithm.selectCoffee

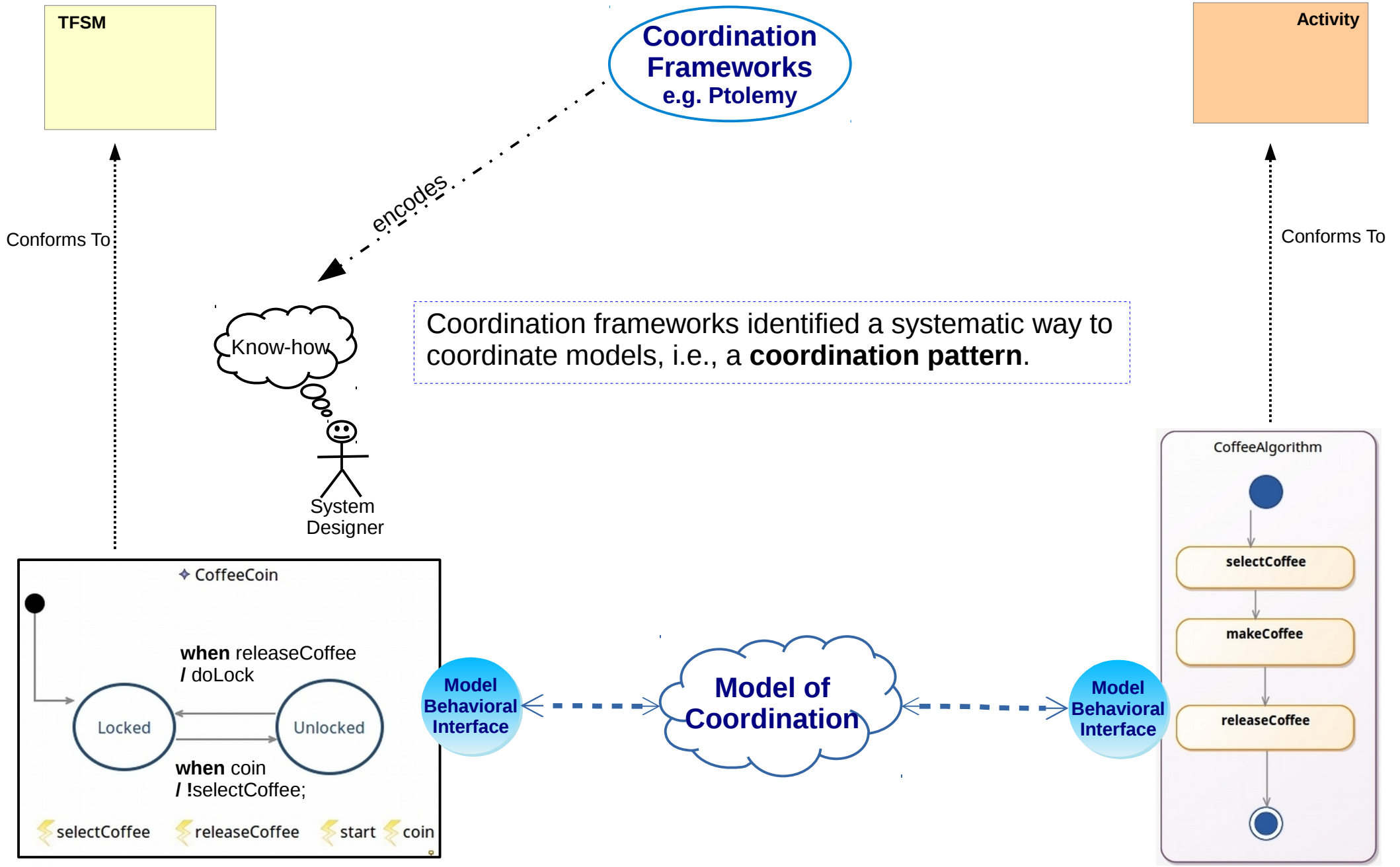



Gemoc

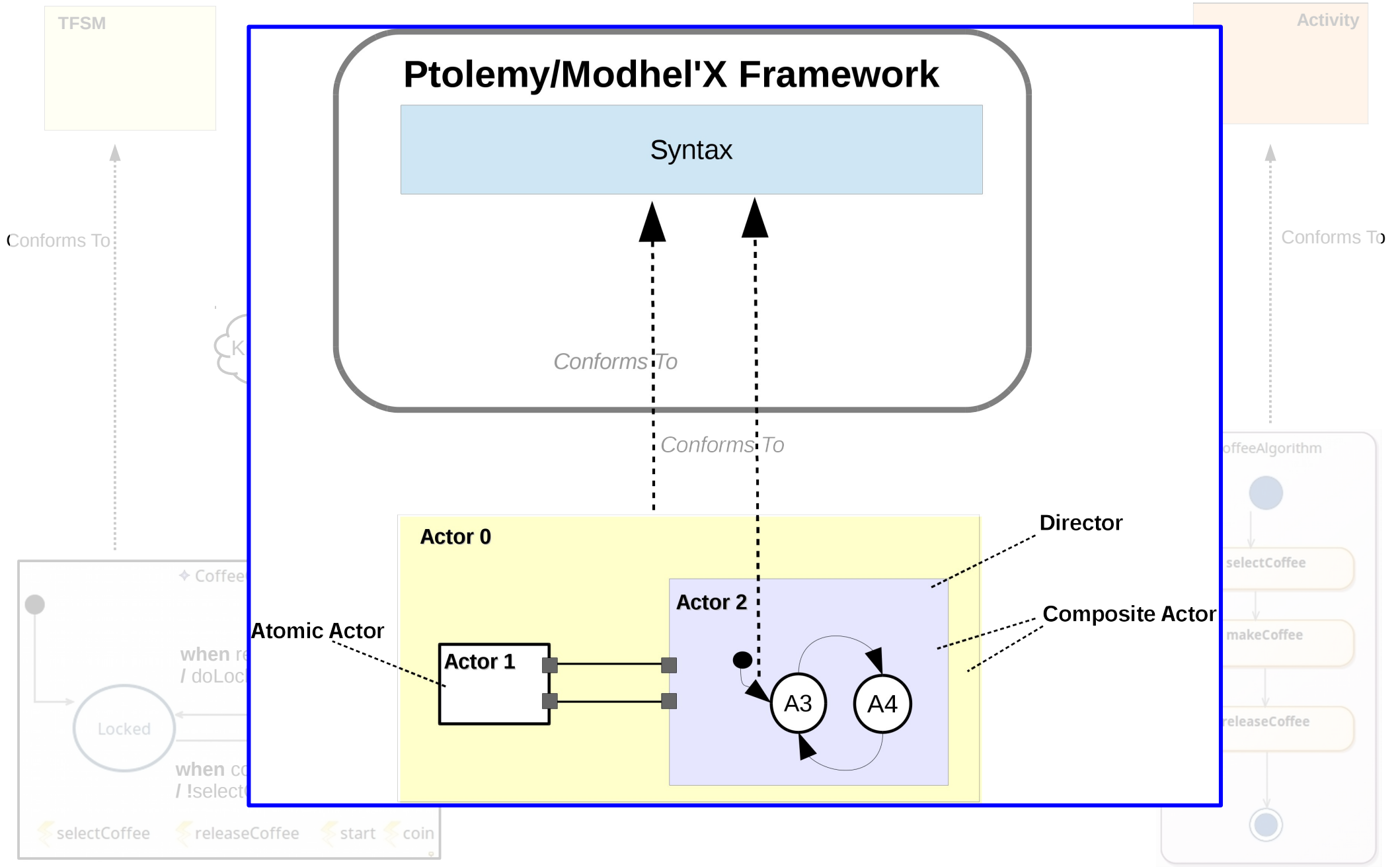
# Coordination Patterns



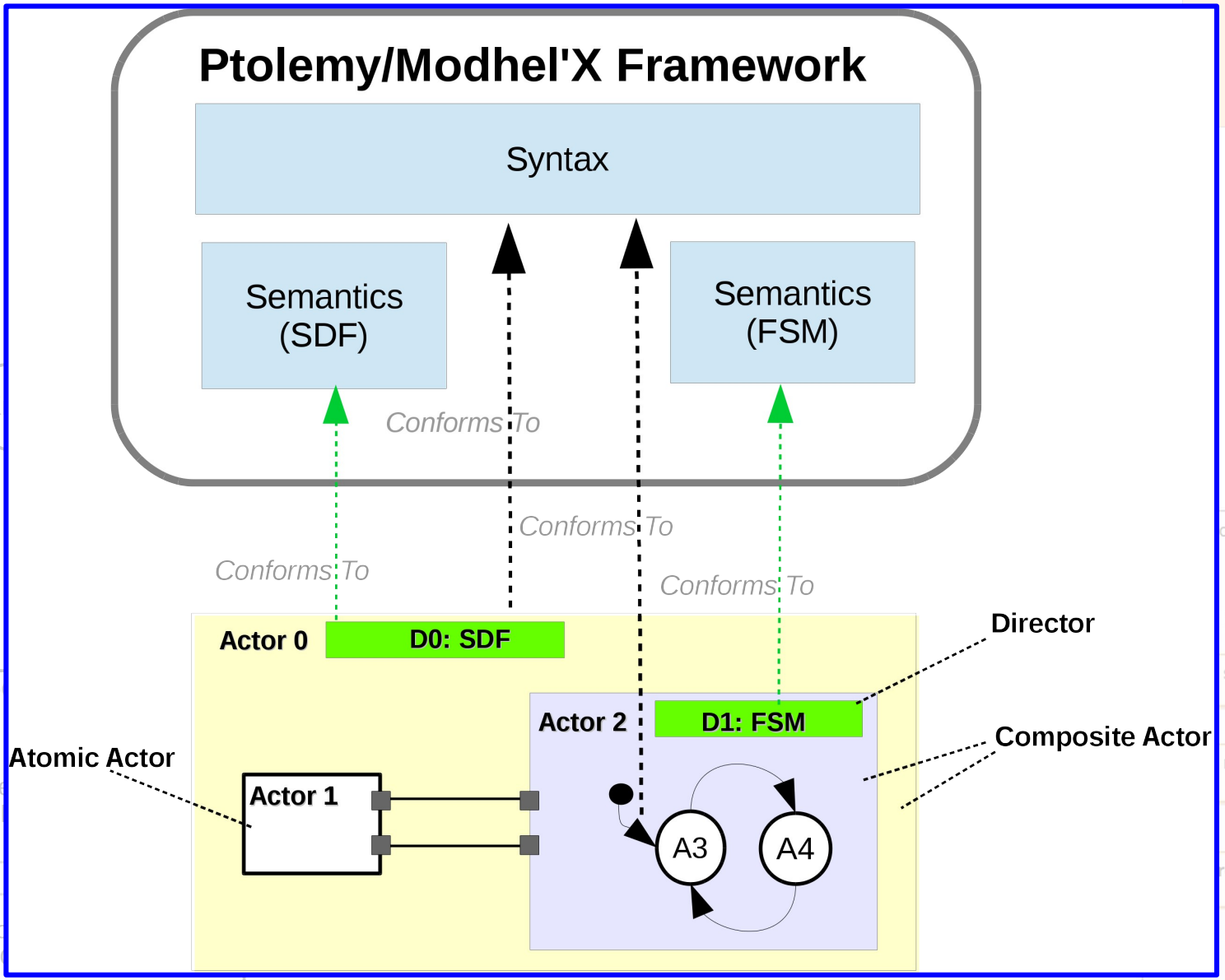
# Coordination Frameworks



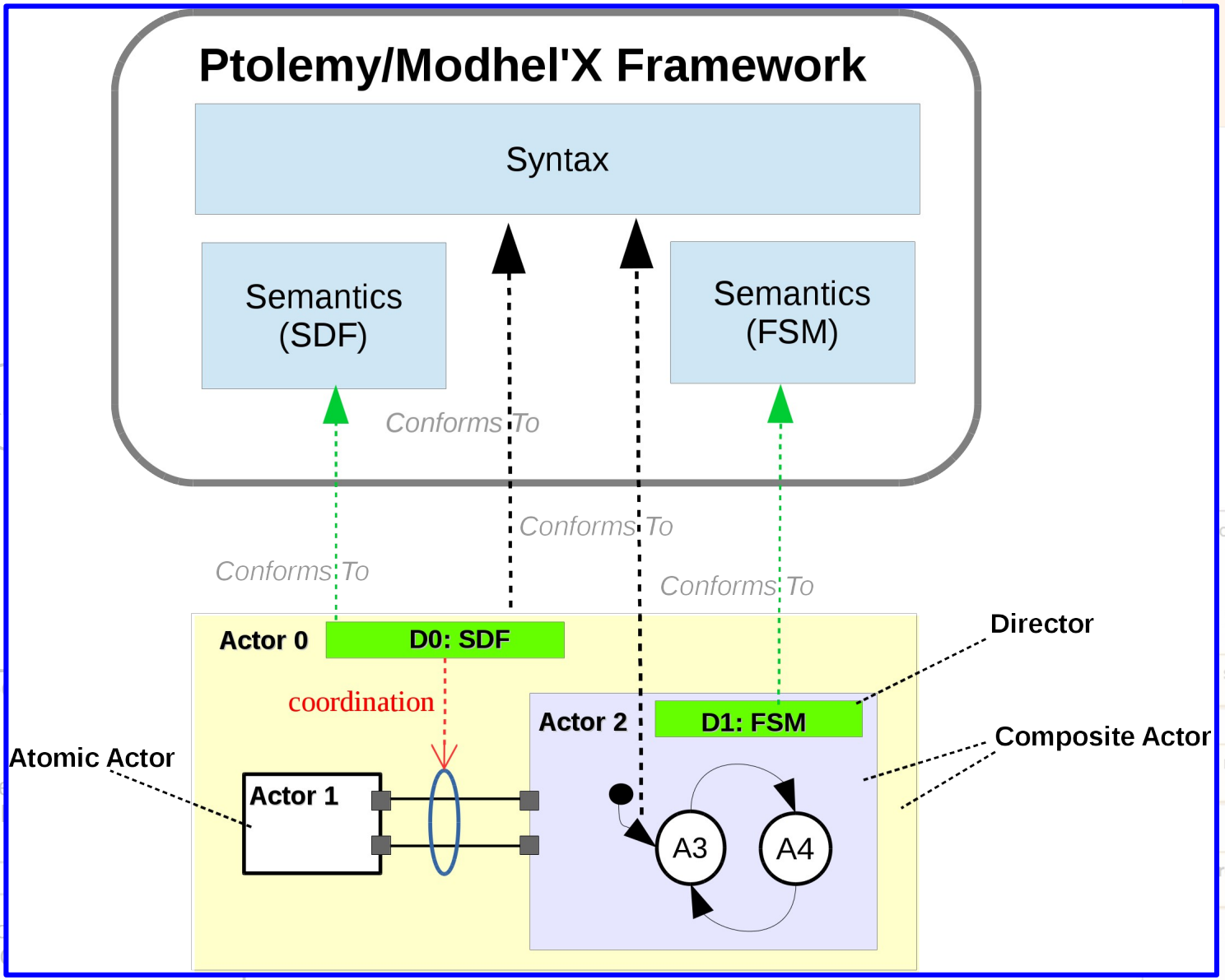
# Coordination Frameworks



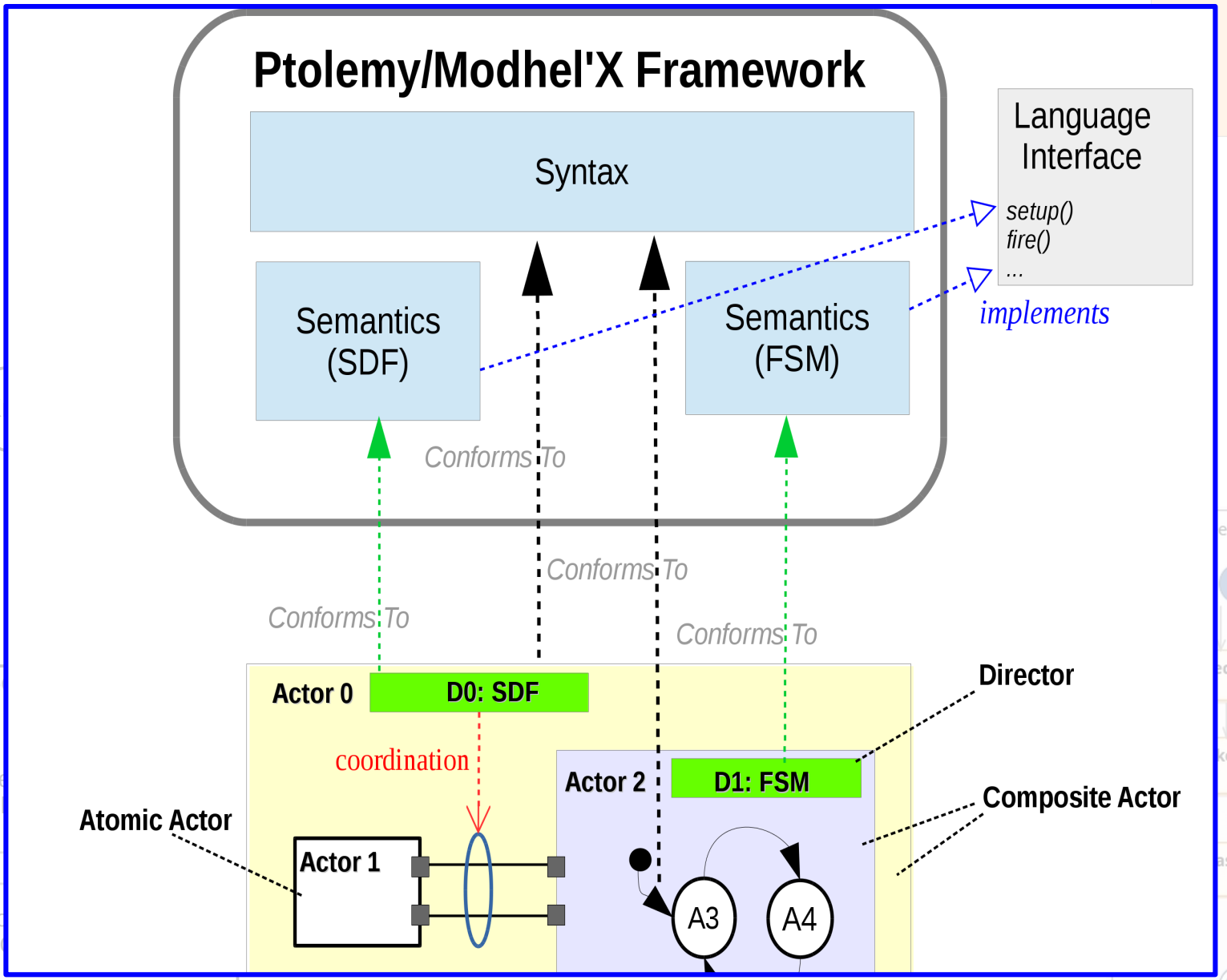
# Coordination Frameworks



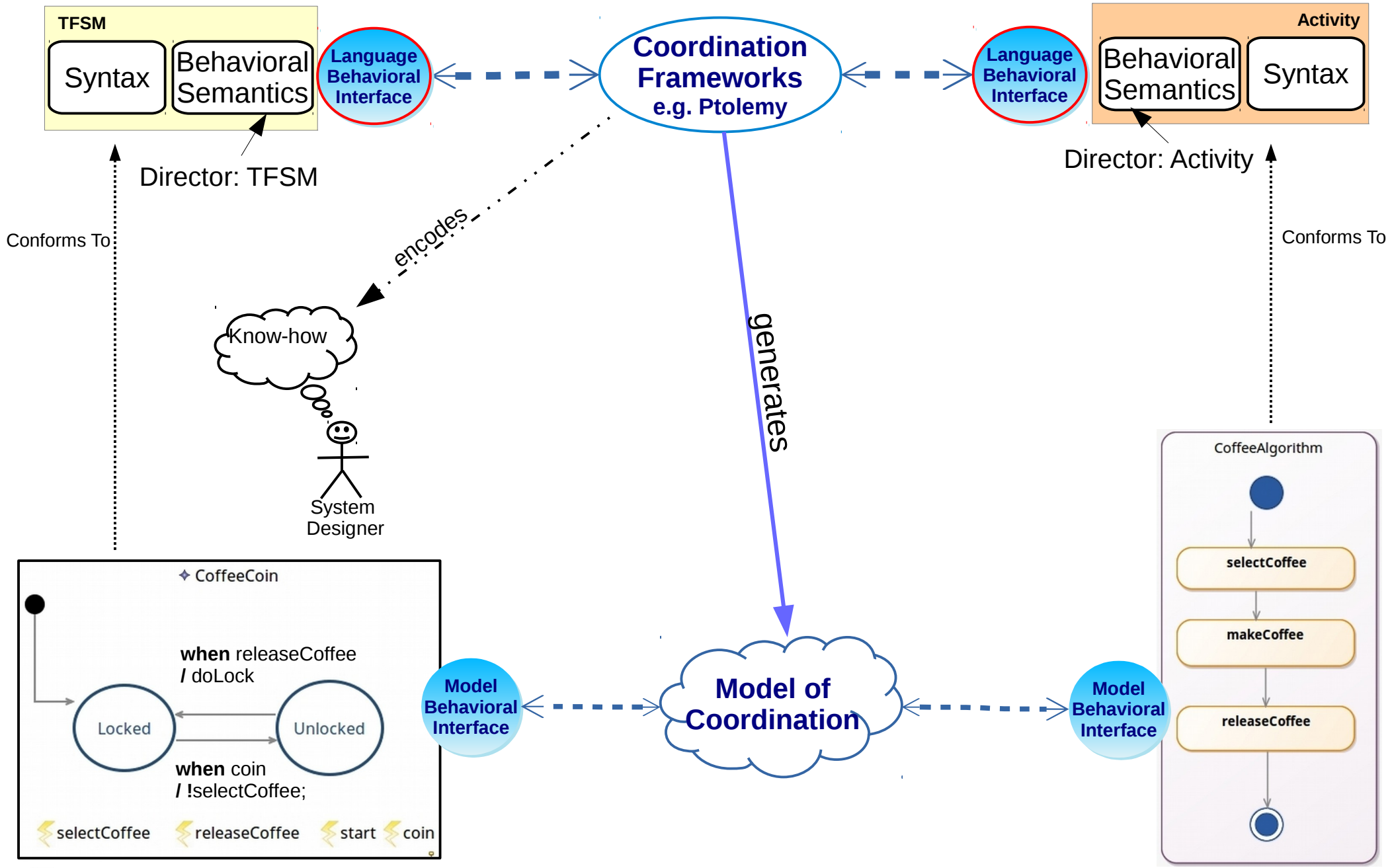
# Coordination Frameworks



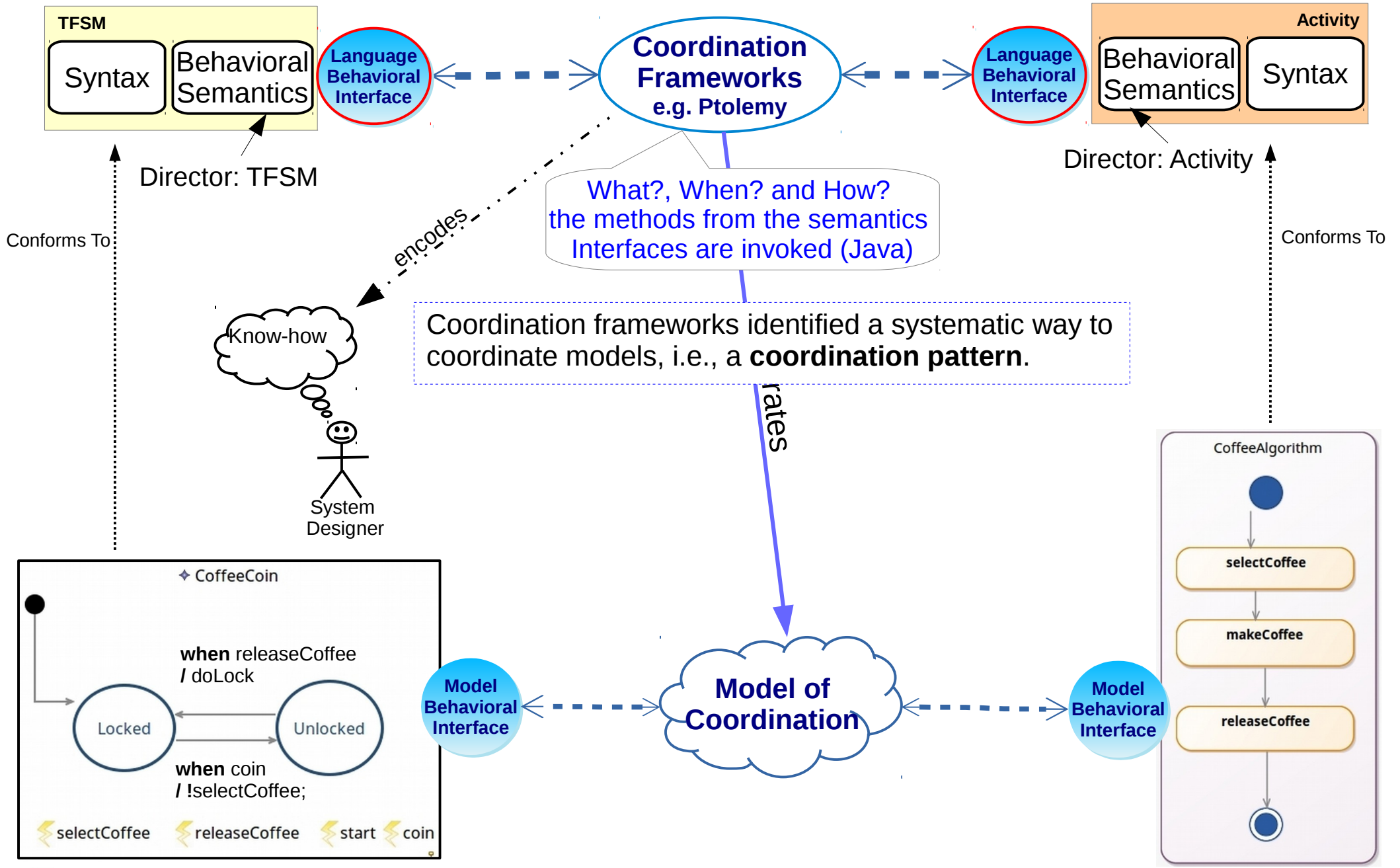
# Coordination Frameworks



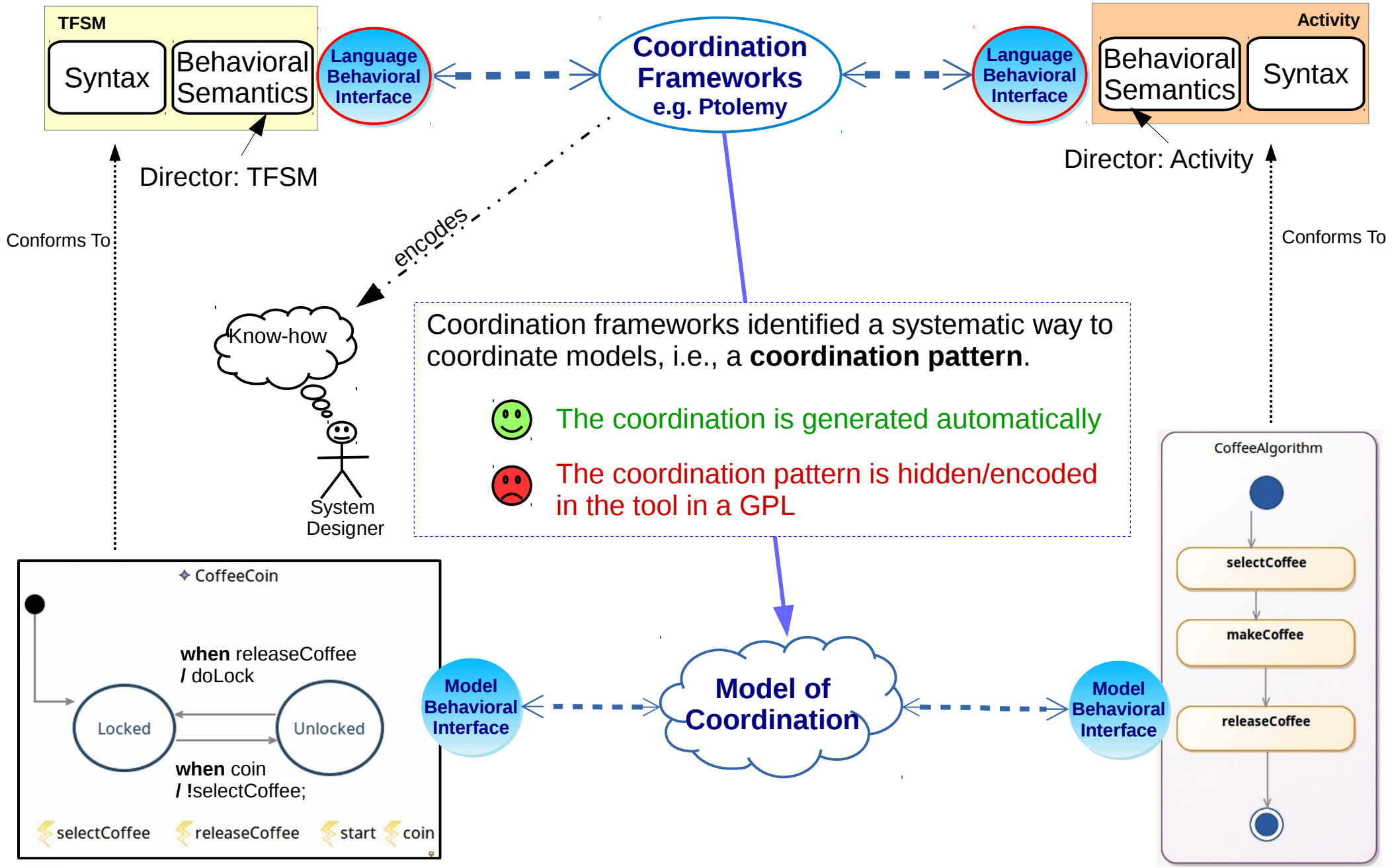
# Coordination Frameworks



# Coordination Frameworks



# Coordination Frameworks

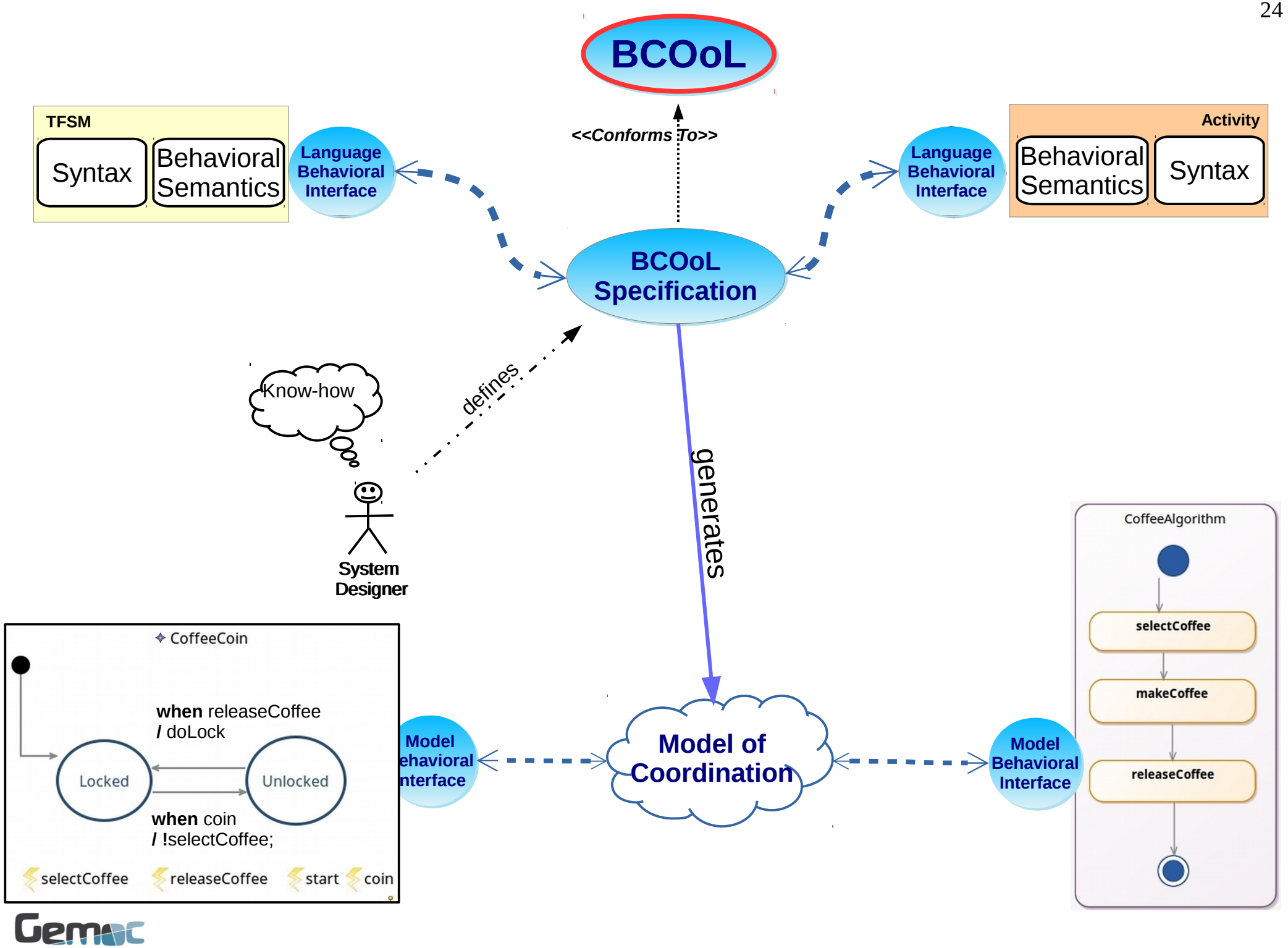


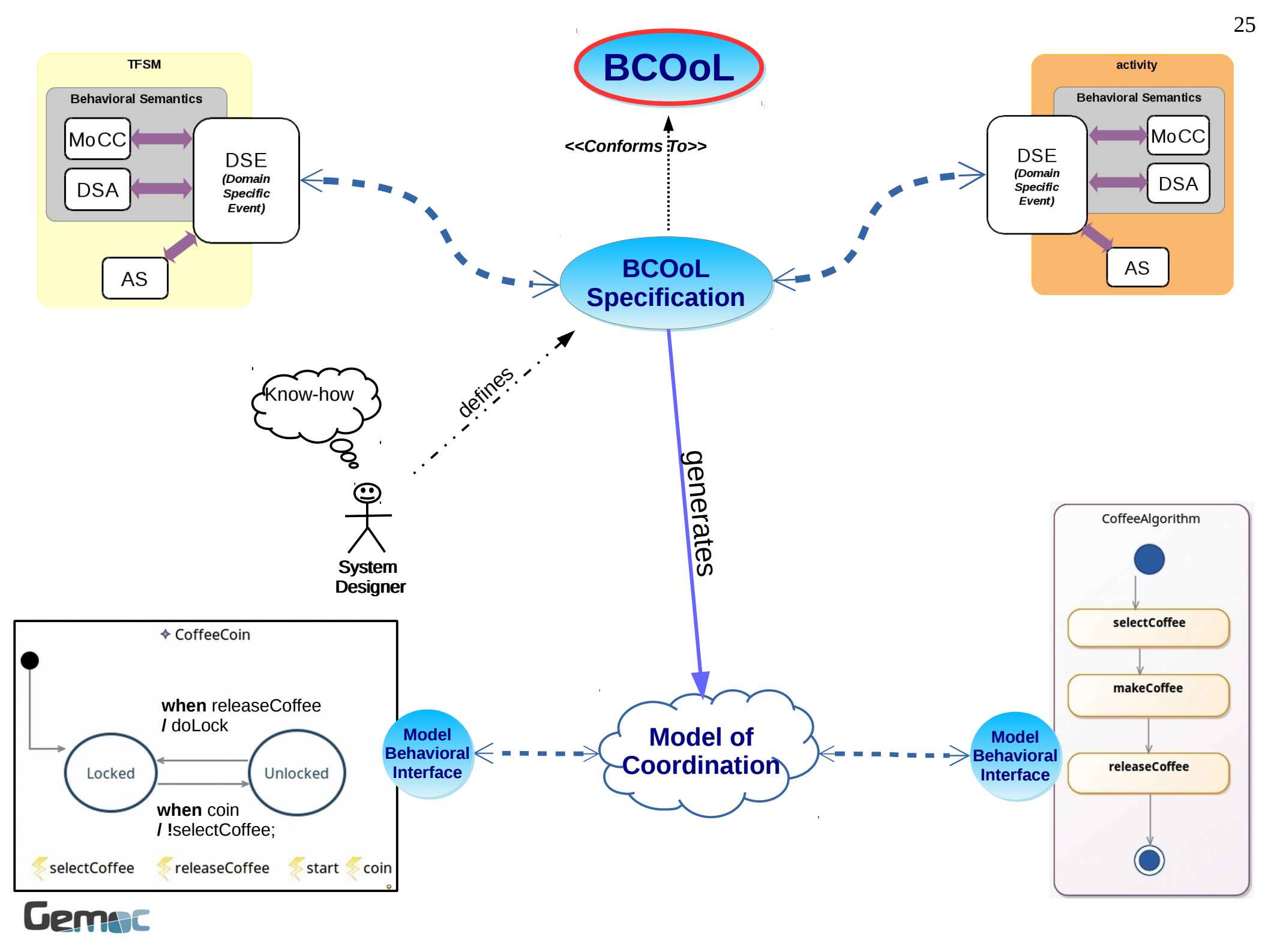
# Take-Away Lessons

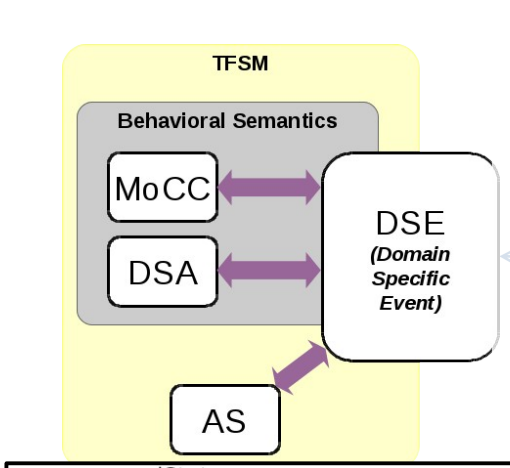
- Coordination Languages & ADLs:
  -  The coordination is modeled explicitly
  -  The coordination is defined rule by rule
- Coordination Frameworks:
  -  The coordination is generated automatically based on a coordination pattern
  -  The coordination pattern is hidden/encoded in the tool in a GPL

# Outline

- State of the Art
  - Coordination Languages
  - Coordination Frameworks
- Our proposal:
  - The **B**ehavioral **C**oordination **O**perat***o***r **L**anguage
- Conclusion


BCOoL
<<Conforms To>>
TFSM
Syntax
Behavioral Semantics
Language Behavioral Interface
Language Behavioral Interface
Activity
Behavioral Semantics
Syntax
BCOoL Specification
Know-how
defines
generates
System Designer
CoffeeCoin
when releaseCoffee
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
Model ehavioral nterface
Model of Coordination
Model Behavioral Interface
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee


BCOoL
<<Conforms To>>
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
BCOoL Specification
Know-how
defines
System Designer
generates
CoffeeCoin
when releaseCoffee
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
Model Behavioral Interface
Model of Coordination
Model Behavioral Interface
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee

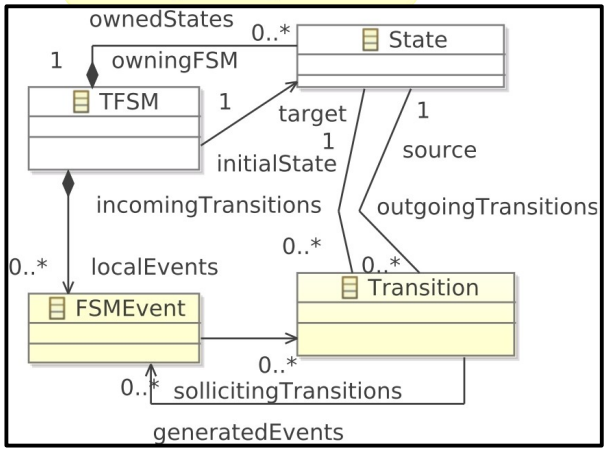



BCOoL

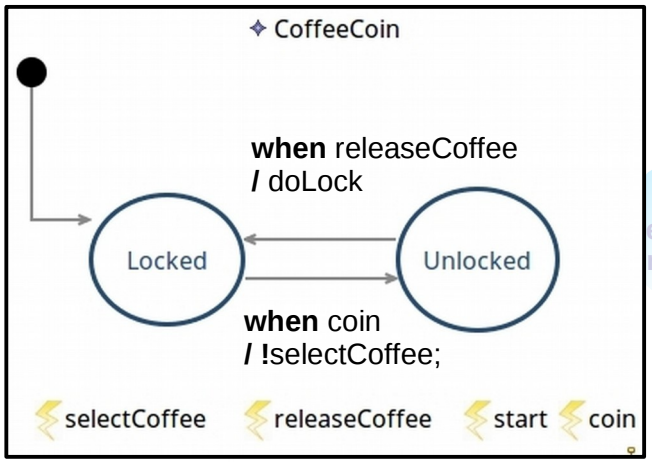
<<Conforms To>>




BCOoL Specification




generates



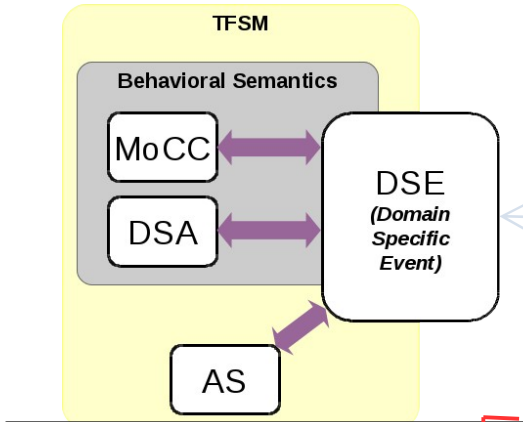

Model Behavioral Interface

Model of Coordination

Model Behavioral Interface




Gemoc

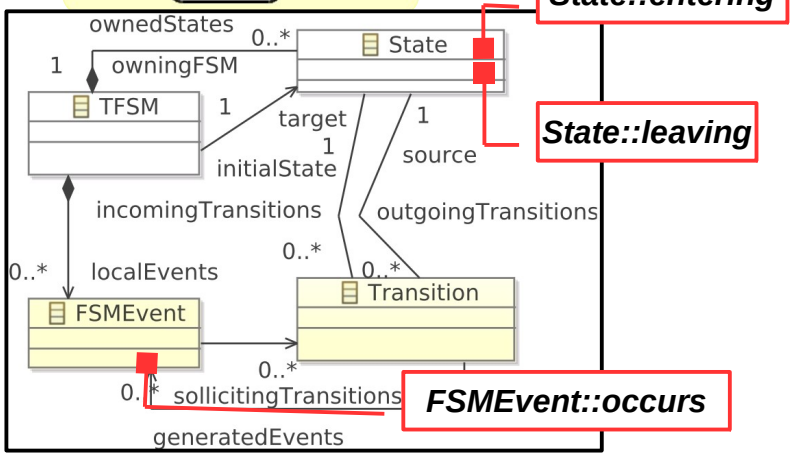



BCOoL

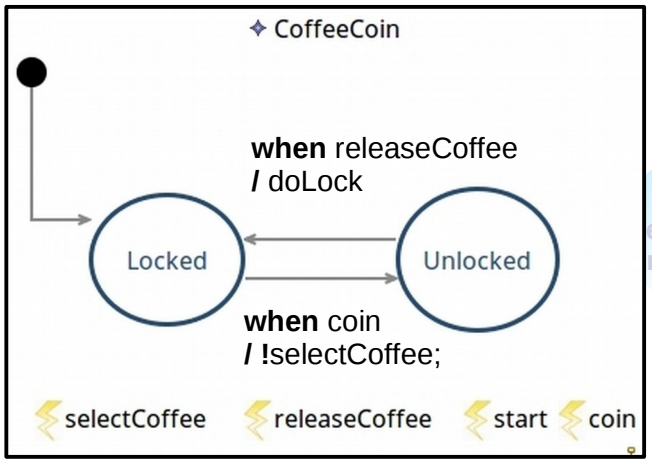
<<Conforms To>>




BCOoL Specification




generates



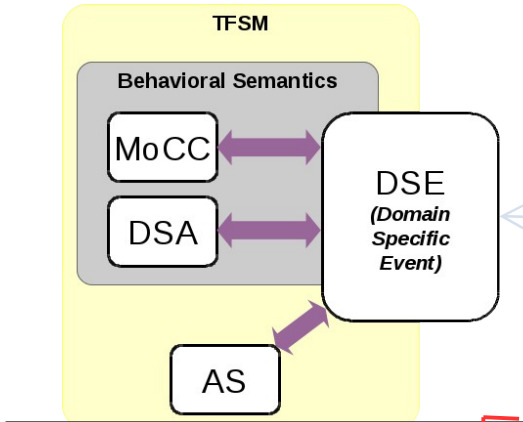

Model Behavioral Interface

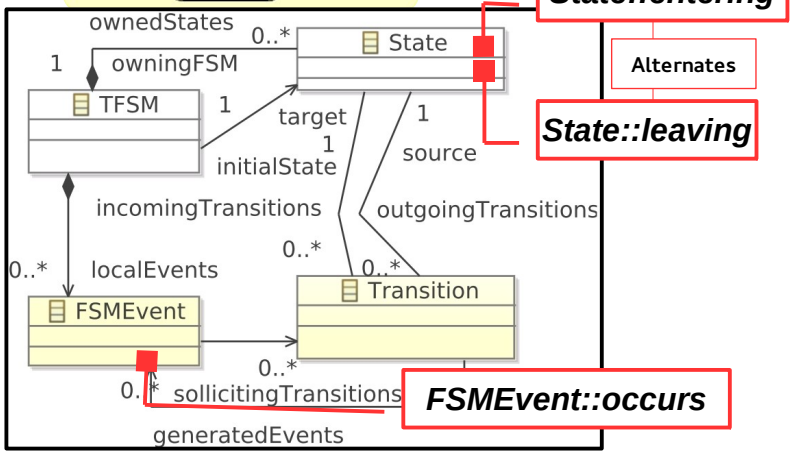
Model of Coordination

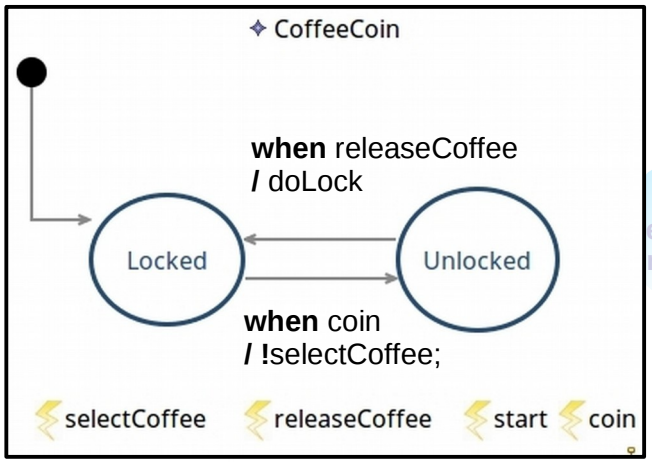
Model Behavioral Interface




Gemoc




BCOoL

<<Conforms To>>

BCOoL Specification

generates

Model of Coordination

Model Behavioral Interface

Model Behavioral Interface

activity

Behavioral Semantics

MoCC

DSA

DSE (Domain Specific Event)

AS

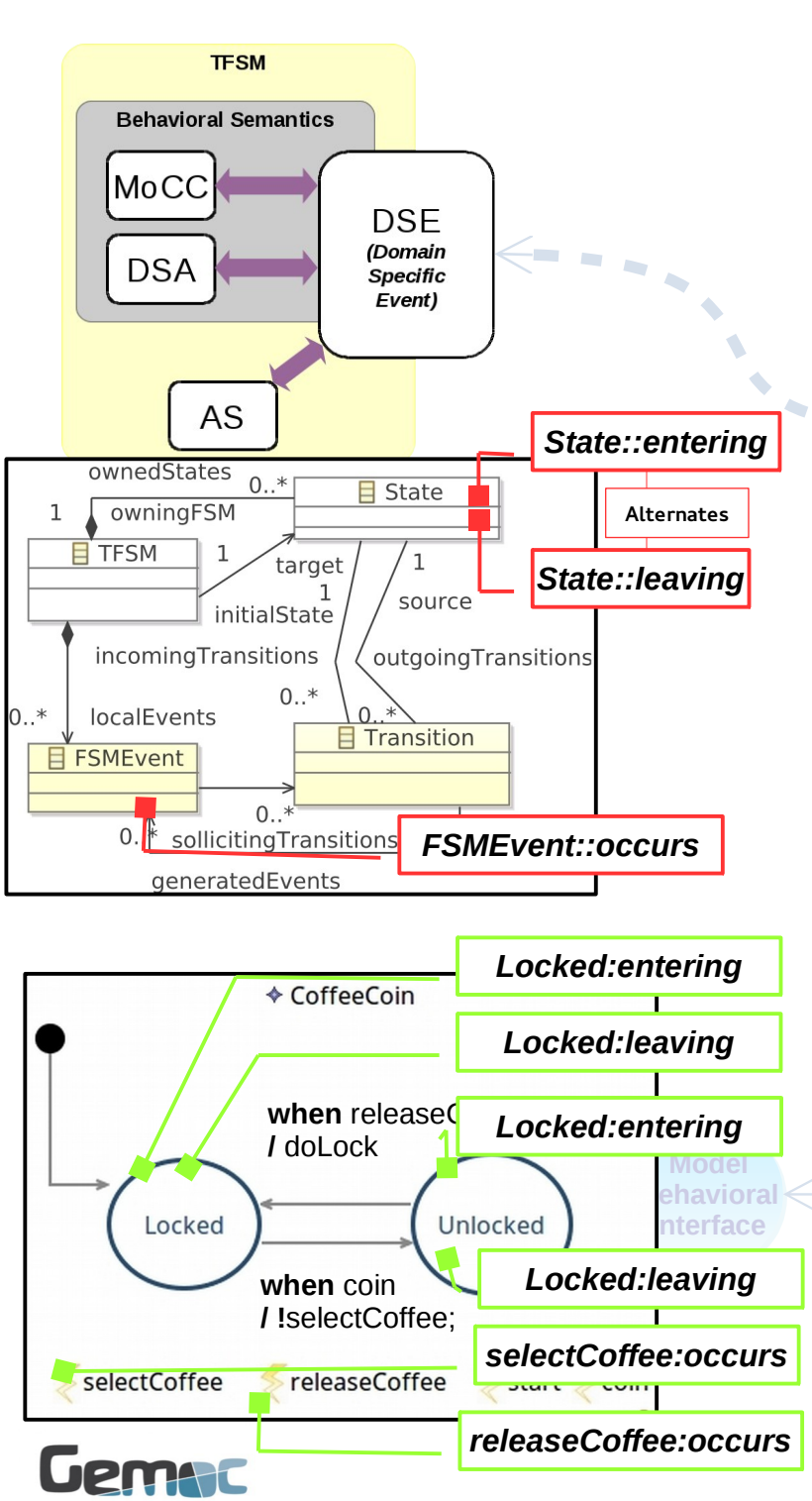
CoffeeAlgorithm

selectCoffee

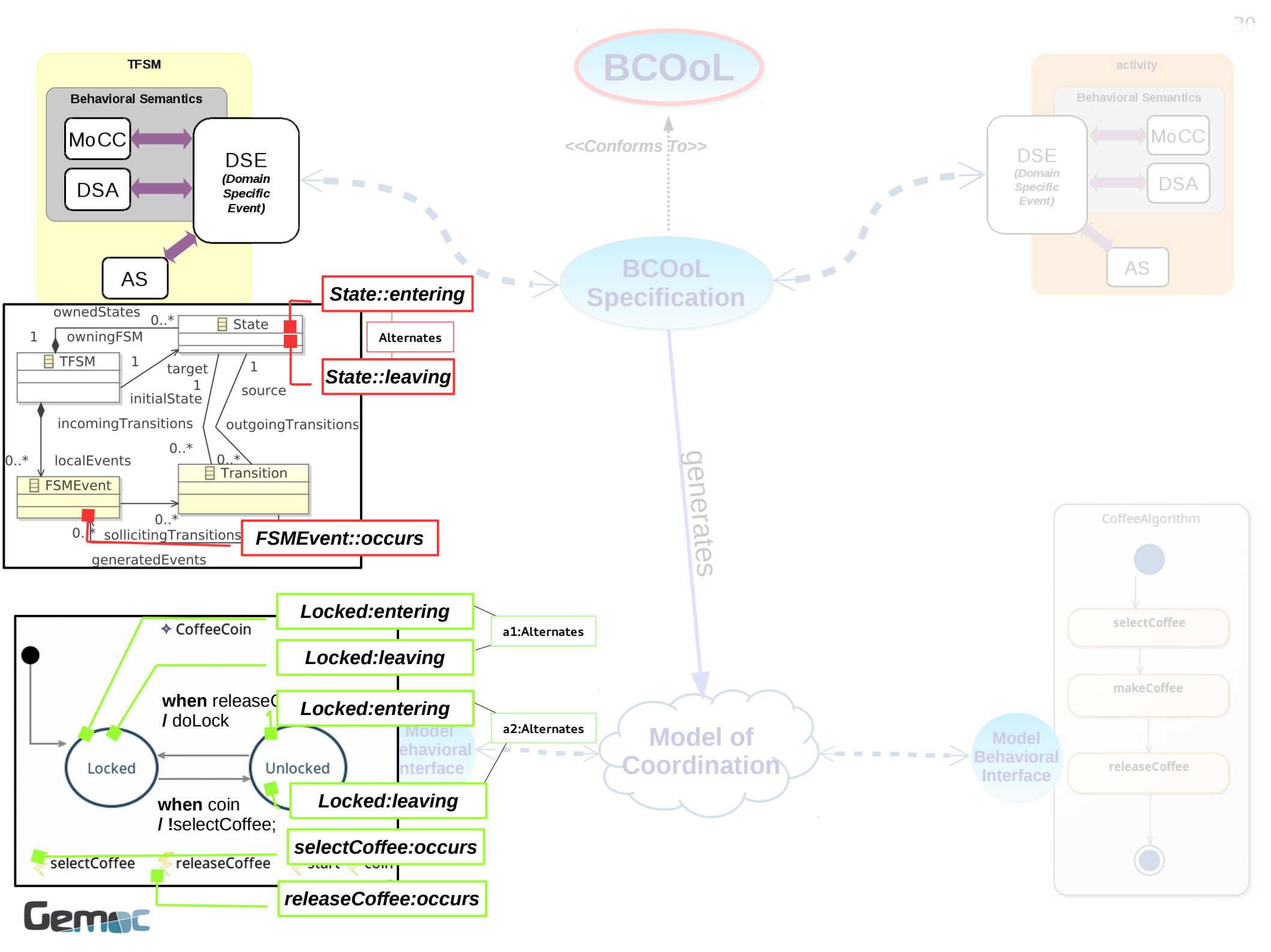
makeCoffee

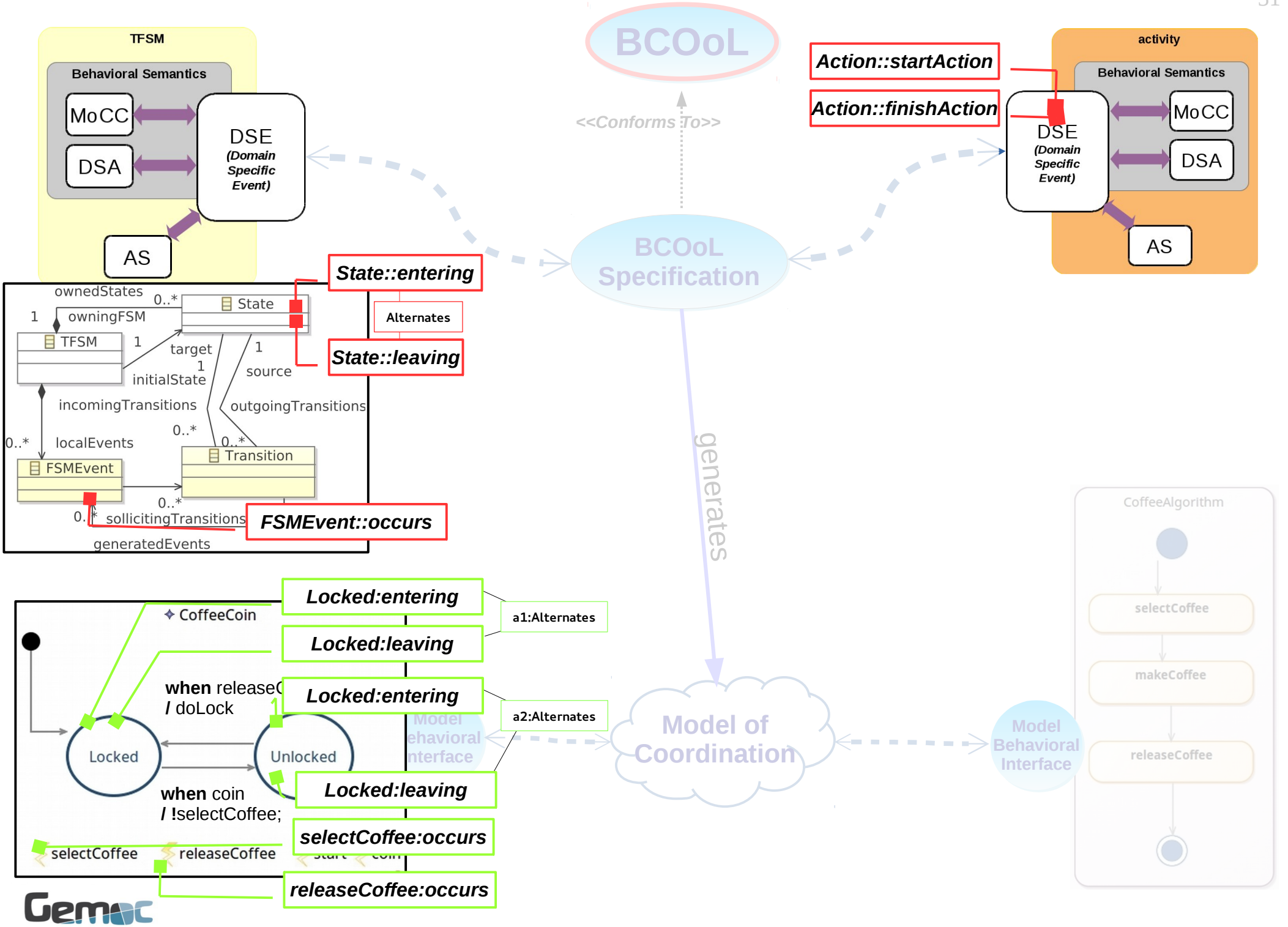
releaseCoffee

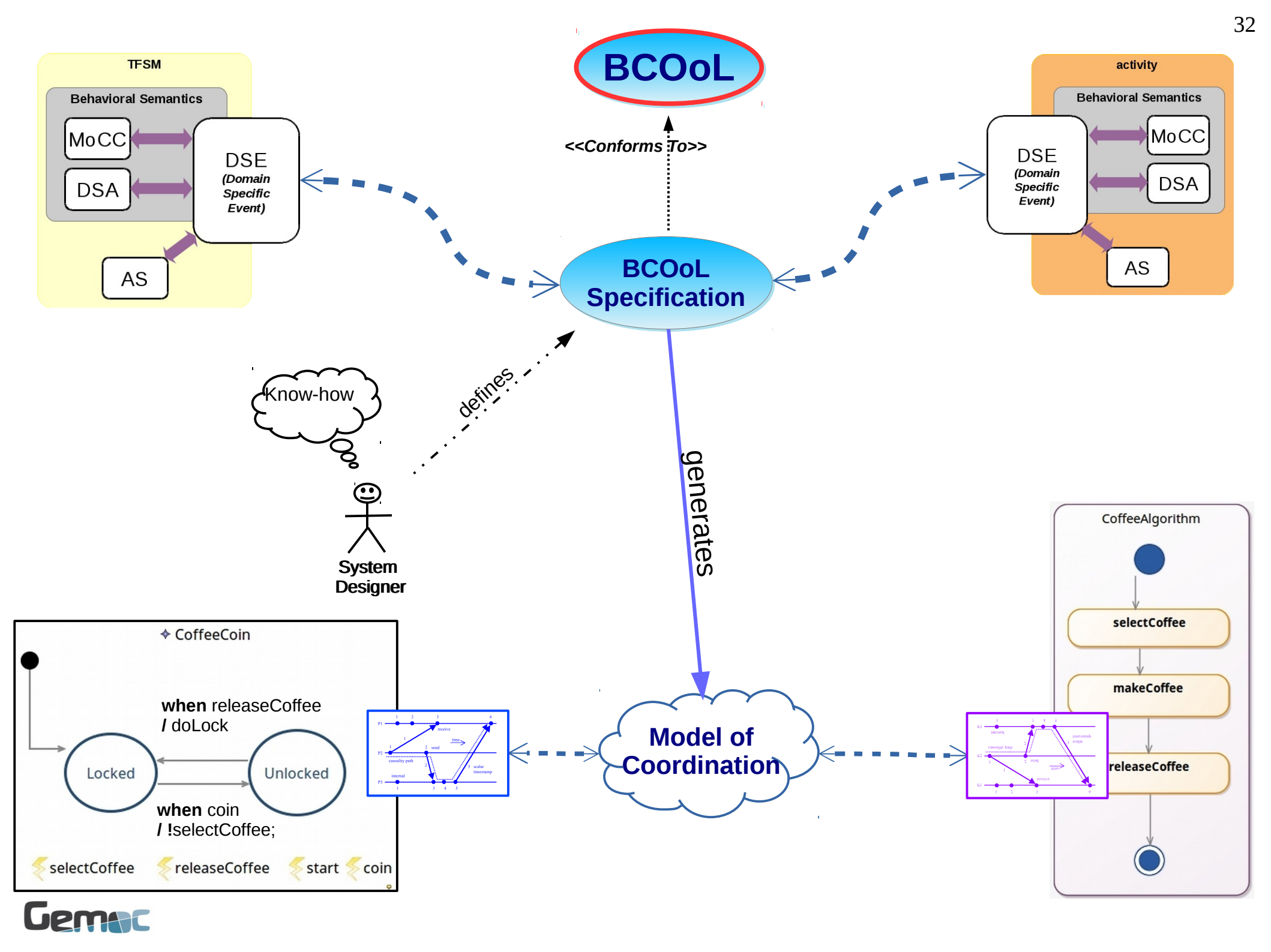
Gemoc

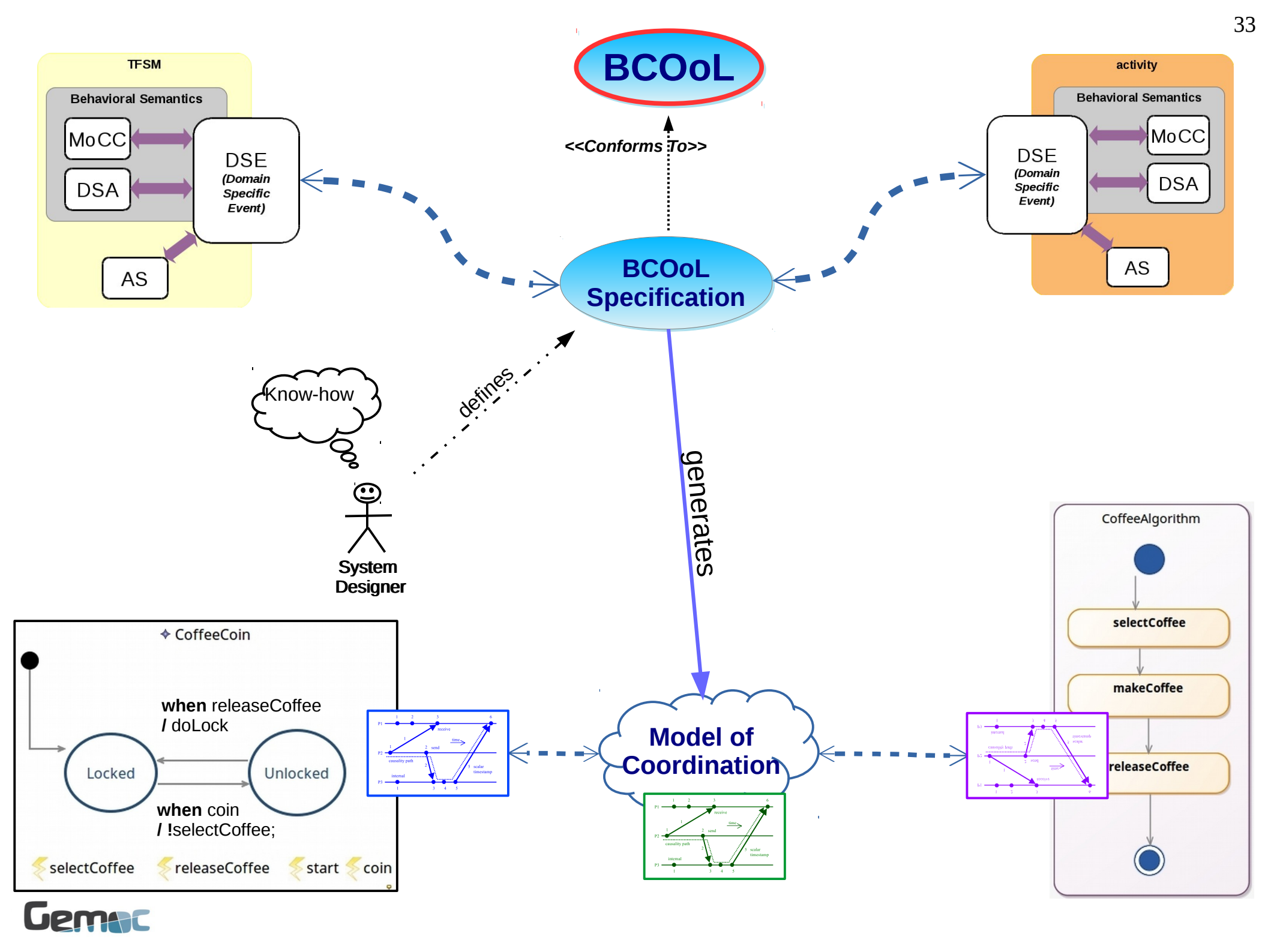

TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
State::entering
Alternates
State::leaving
ownedStates
0..*
State
1
owningFSM
TFSM
1
target
1
initialState
1
source
incomingTransitions
outgoingTransitions
0..*
localEvents
0..*
0..*
FSMEvent
Transition
0..*
0..*
sollicitingTransitions
FSMEvent::occurs
generatedEvents
Locked:entering
CoffeeCoin
Locked:leaving
when releaseC
/ doLock
Locked:entering
Locked
Unlocked
when coin
/ !selectCoffee;
Locked:leaving
selectCoffee:occurs
selectCoffee
releaseCoffee
releaseCoffee:occurs
Gemoc

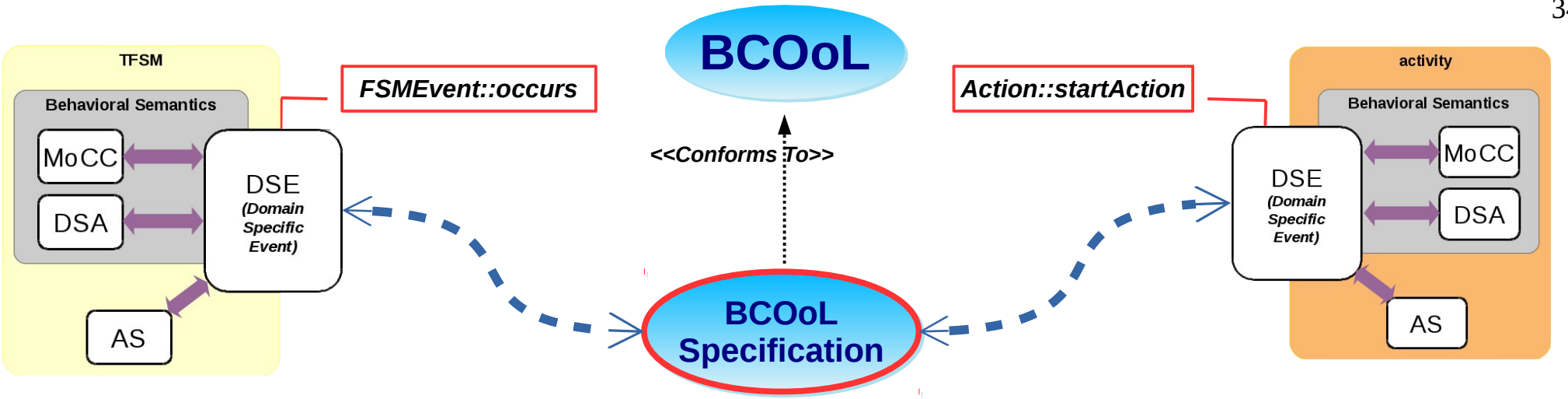

BCOoL
<<Conforms To>>
BCOoL Specification
generates
Model of Coordination
Model Behavioral Interface
activity
Behavioral Semantics
DSE
(Domain Specific Event)
MoCC
DSA
AS
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee


TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
BCOoL
<<Conforms To>>
BCOoL Specification
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
State::entering
Alternates
State::leaving
ownedStates
0..*
State
1
owningFSM
TFSM
1
target
1
initialState
1
source
incomingTransitions
outgoingTransitions
0..*
localEvents
0..*
FSMEvent
Transition
0..*
0..*
sollicitingTransitions
FSMEvent::occurs
generatedEvents
generates
Locked:entering
a1:Alternates
Locked:leaving
CoffeeCoin
when releaseC
/ doLock
Locked:entering
a2:Alternates
Locked
Unlocked
Model Behavioral Interface
Model of Coordination
Model Behavioral Interface
when coin
/ !selectCoffee;
Locked:leaving
selectCoffee:occurs
selectCoffee
releaseCoffee
releaseCoffee:occurs
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee
Gemoc


BCOoL
<<Conforms To>>
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
BCOoL Specification
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
Action::startAction
Action::finishAction
State::entering
Alternates
State::leaving
ownedStates
0..*
State
1
owningFSM
TFSM
1
target
1
initialState
1
source
incomingTransitions
outgoingTransitions
0..*
0..*
localEvents
0..*
FSMEvent
Transition
0..*
0..*
sollicitingTransitions
FSMEvent::occurs
generatedEvents
generates
Locked:entering
a1:Alternates
Locked:leaving
CoffeeCoin
when releaseC
/ doLock
Locked:entering
a2:Alternates
Locked
Unlocked
Model Behavioral Interface
Model of Coordination
Model Behavioral Interface
when coin
/ !selectCoffee;
Locked:leaving
selectCoffee:occurs
selectCoffee
releaseCoffee
releaseCoffee:occurs
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee
Gemoc

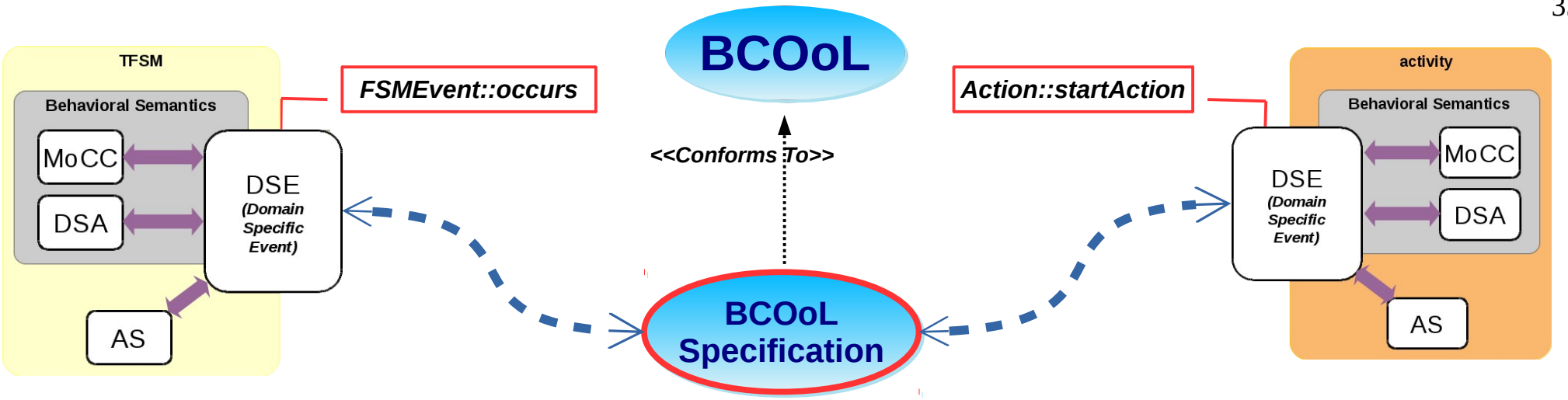

BCOoL
<<Conforms To>>
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
BCOoL Specification
Know-how
defines
System Designer
generates
CoffeeCoin
when releaseCoffee
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
Model of Coordination
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee


BCOoL
<<Conforms To>>
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
BCOoL Specification
Know-how
defines
System Designer
generates
CoffeeCoin
when releaseCoffee
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
Model of Coordination
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee

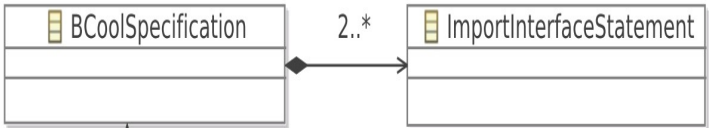



BCOoL Metamodel

SyncFSMEventsAndActions.bcool

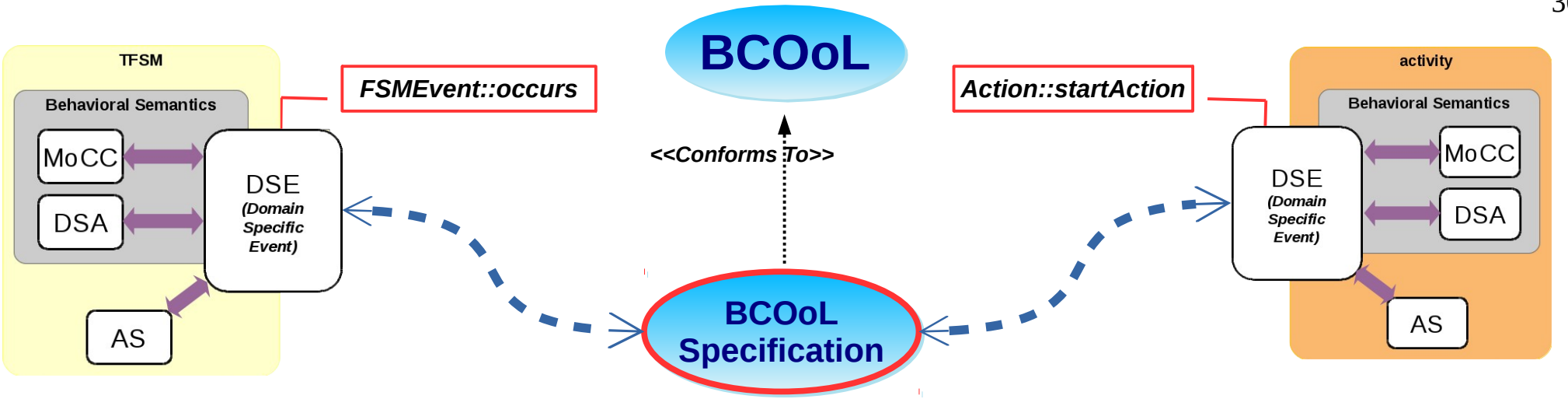



## BCOoL Metamodel

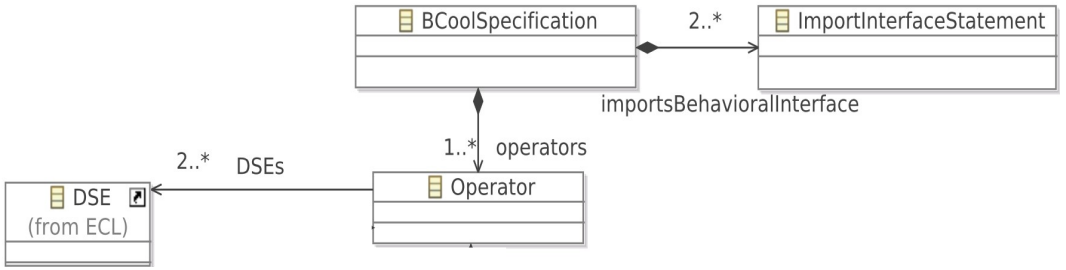



## SyncFSMEventsAndActions.bcool

```
ImportInterface tfsm;
ImportInterface Activity;
```

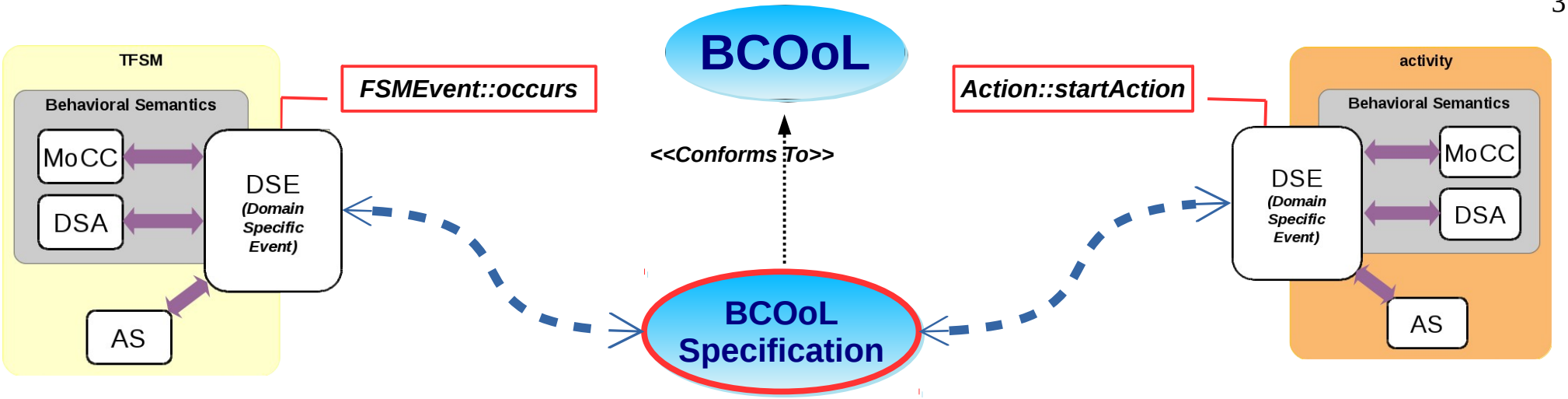



## BCOoL Metamodel

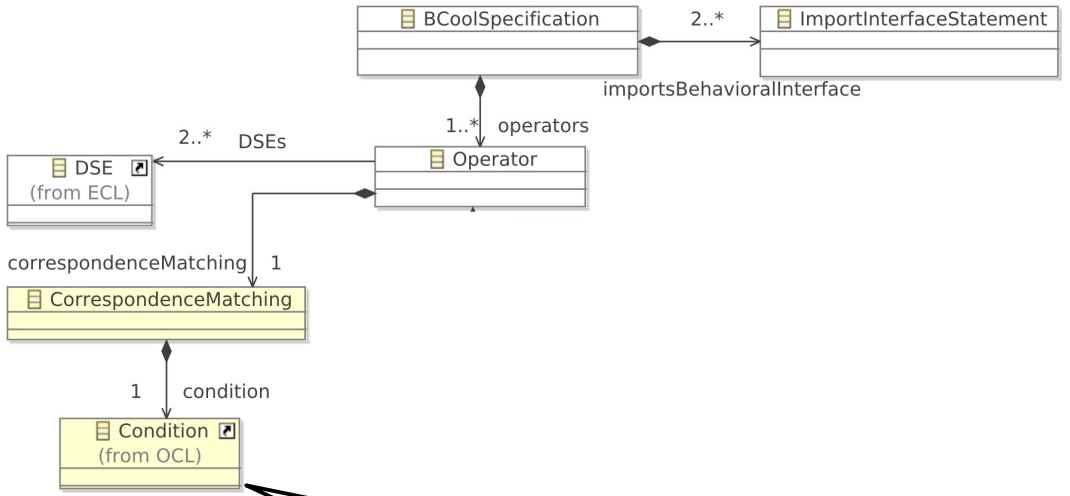



## SyncFSMEventsAndActions.bcool

```
ImportInterface tfsm;
ImportInterface Activity;

Operator RendezVousWhenSameName
(FSMEvent::occurs,Action::startAction)



End Operator;
```



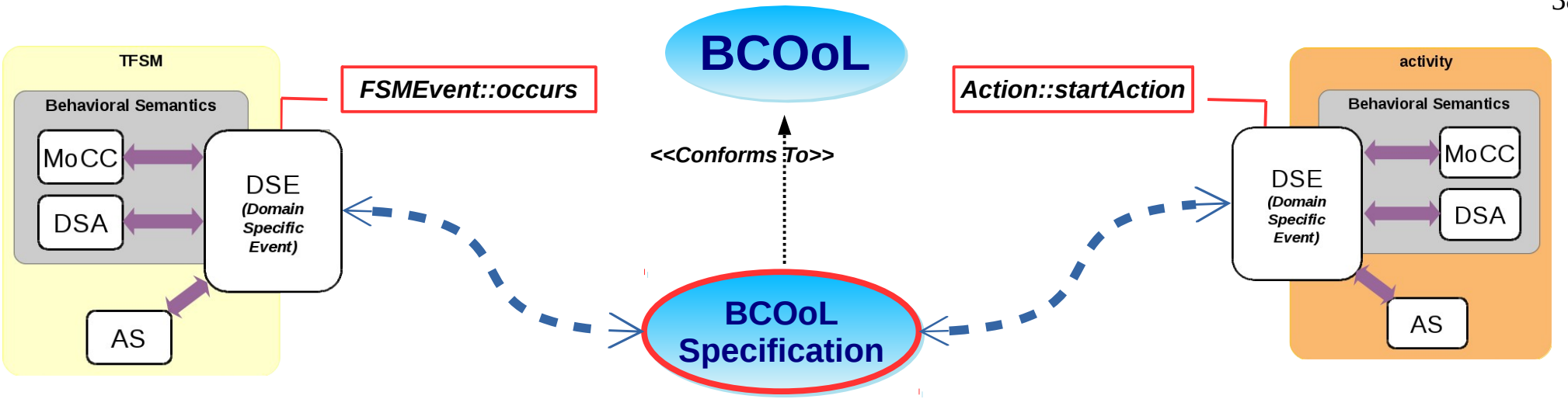

## BCOoL Metamodel



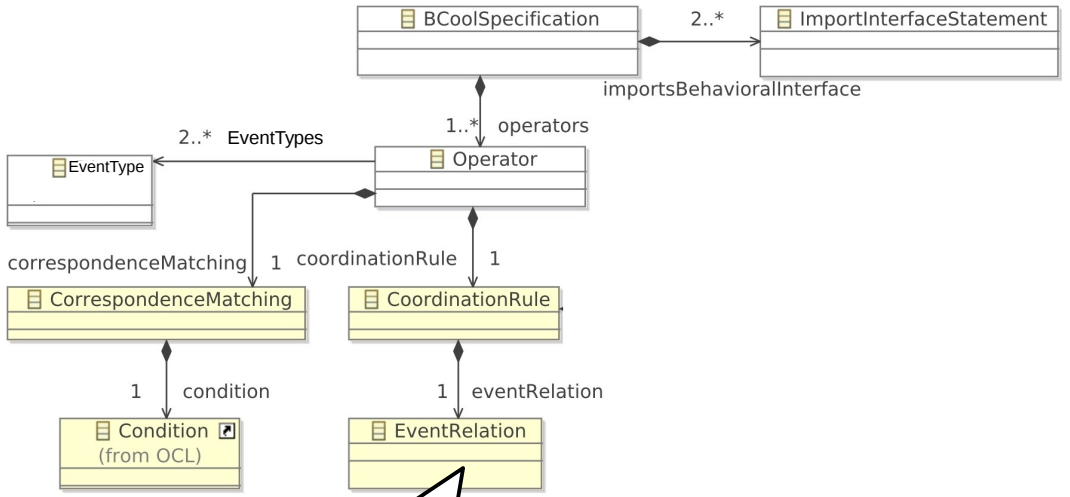

OCL Boolean Expression between model elements

## SyncFSMEventsAndActions.bcool

```
ImportInterface tfsm;
ImportInterface Activity;

Operator RendezVousWhenSameName
(FSMEvent::occurs,Action::startAction)

  When(occurs.name = startAction.name);



End Operator;
```



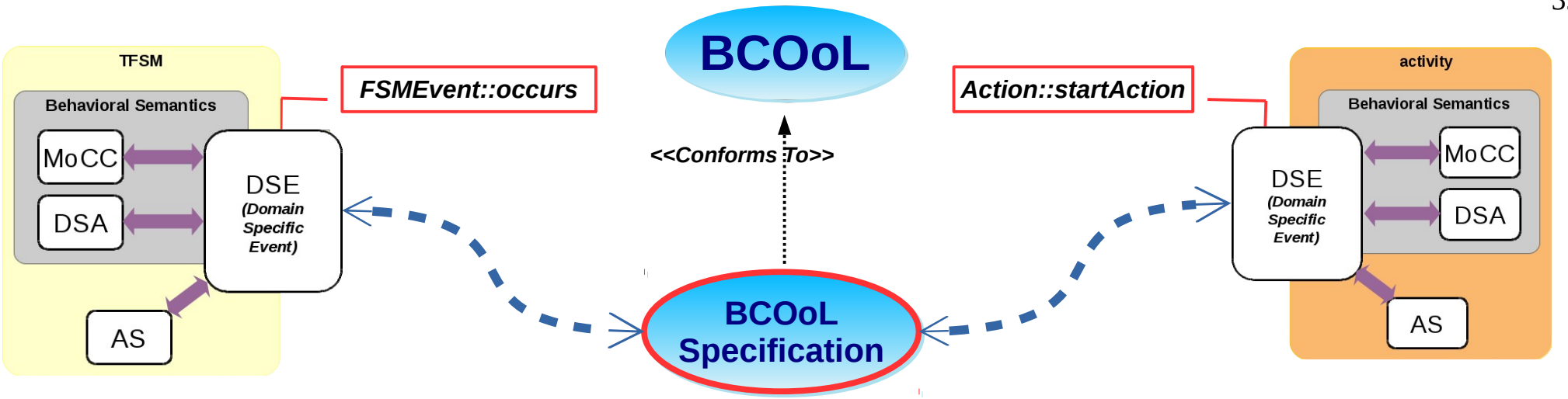

## BCOoL Metamodel



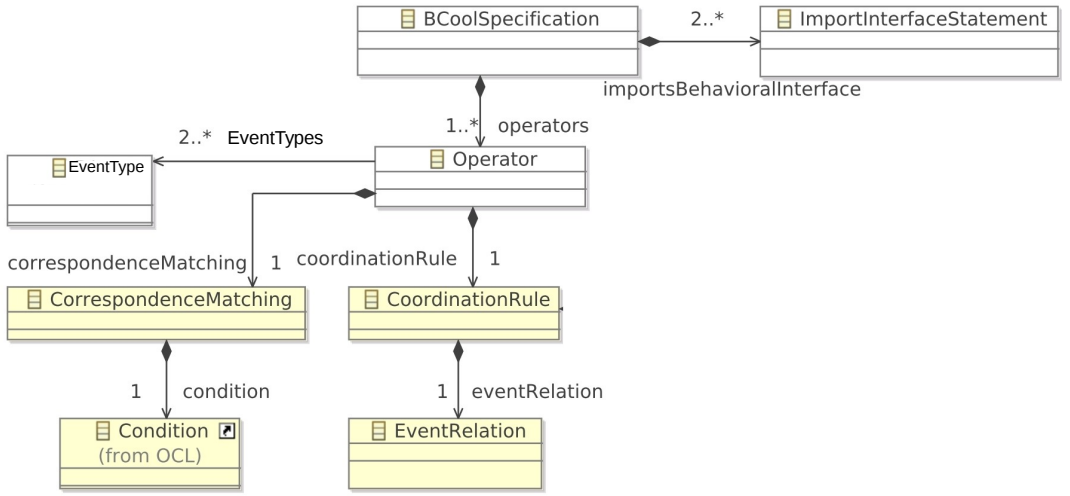

Causal and Temporal relations between Events e.g., Rendezvous, Precedes, etc.

## SyncFSMEventsAndActions.bcool

```
ImportInterface tfsm;
ImportInterface Activity;

Operator RendezVousWhenSameName
(FSMEvent::occurs,Action::startAction)

  When(occurs.name = startAction.name);

  CoordinationRule:
     RendezVous (occurs, startAction)
End Operator;
```




## BCOoL Metamodel

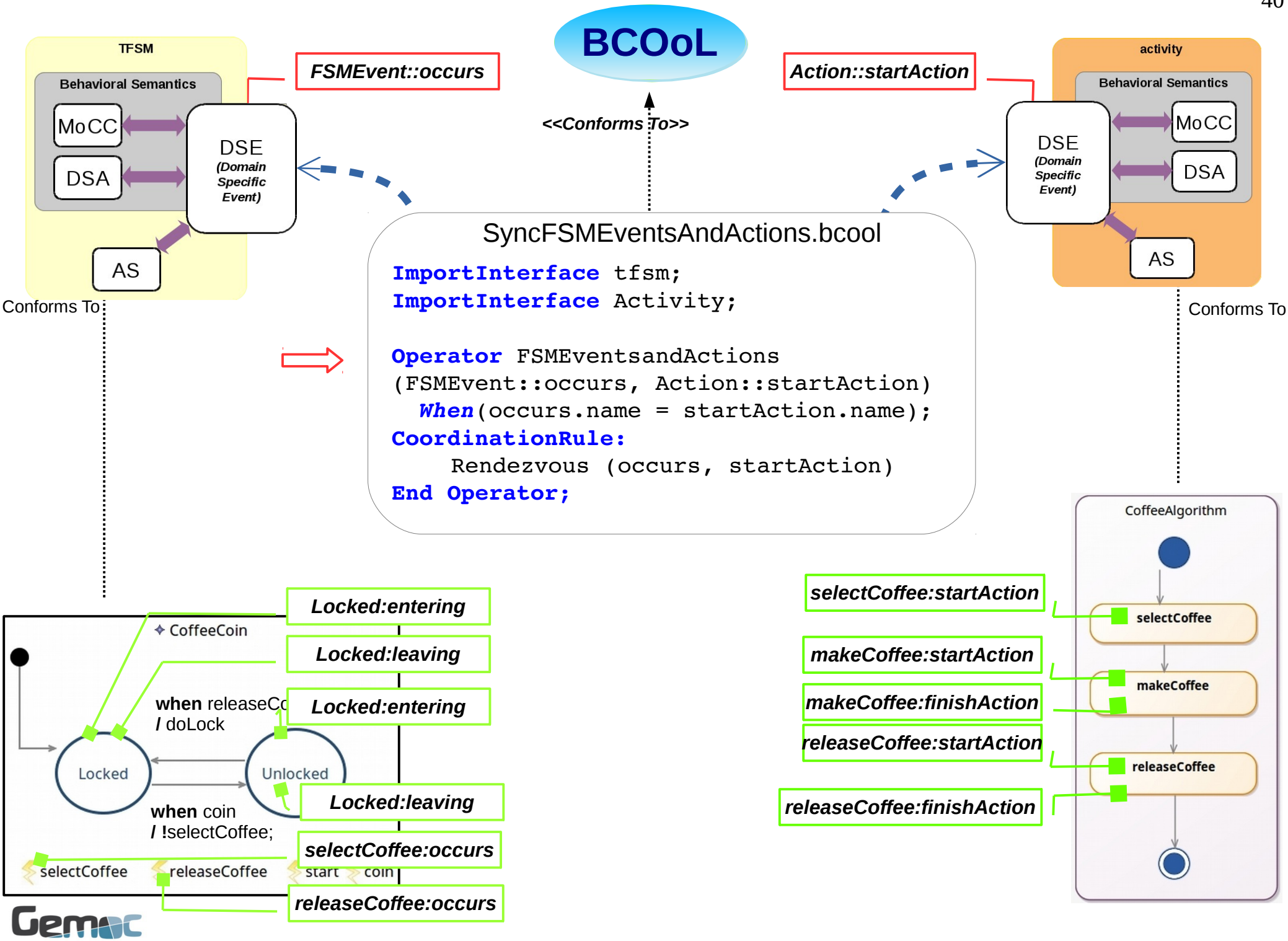


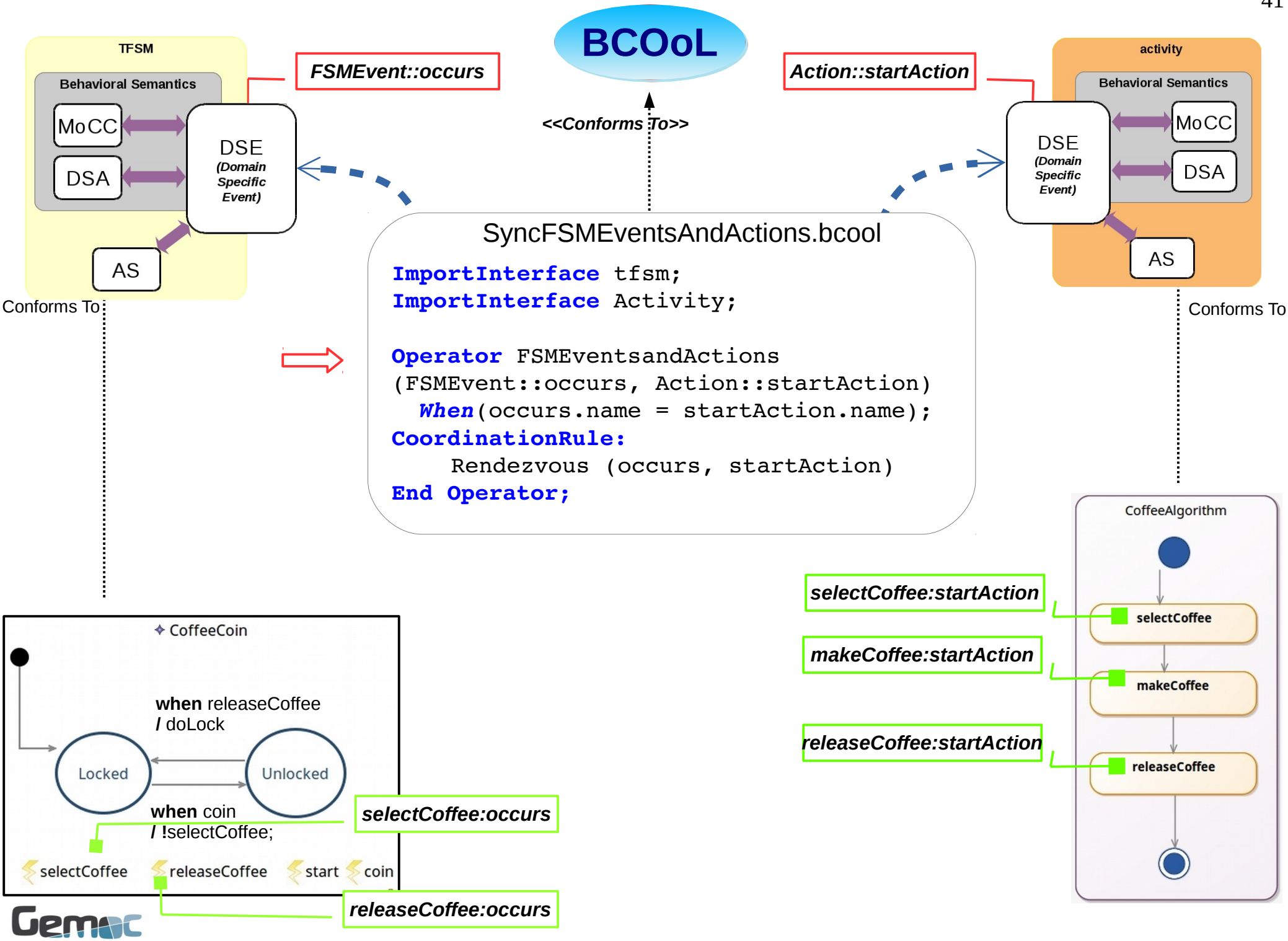

## SyncFSMEventsAndActions.bcool

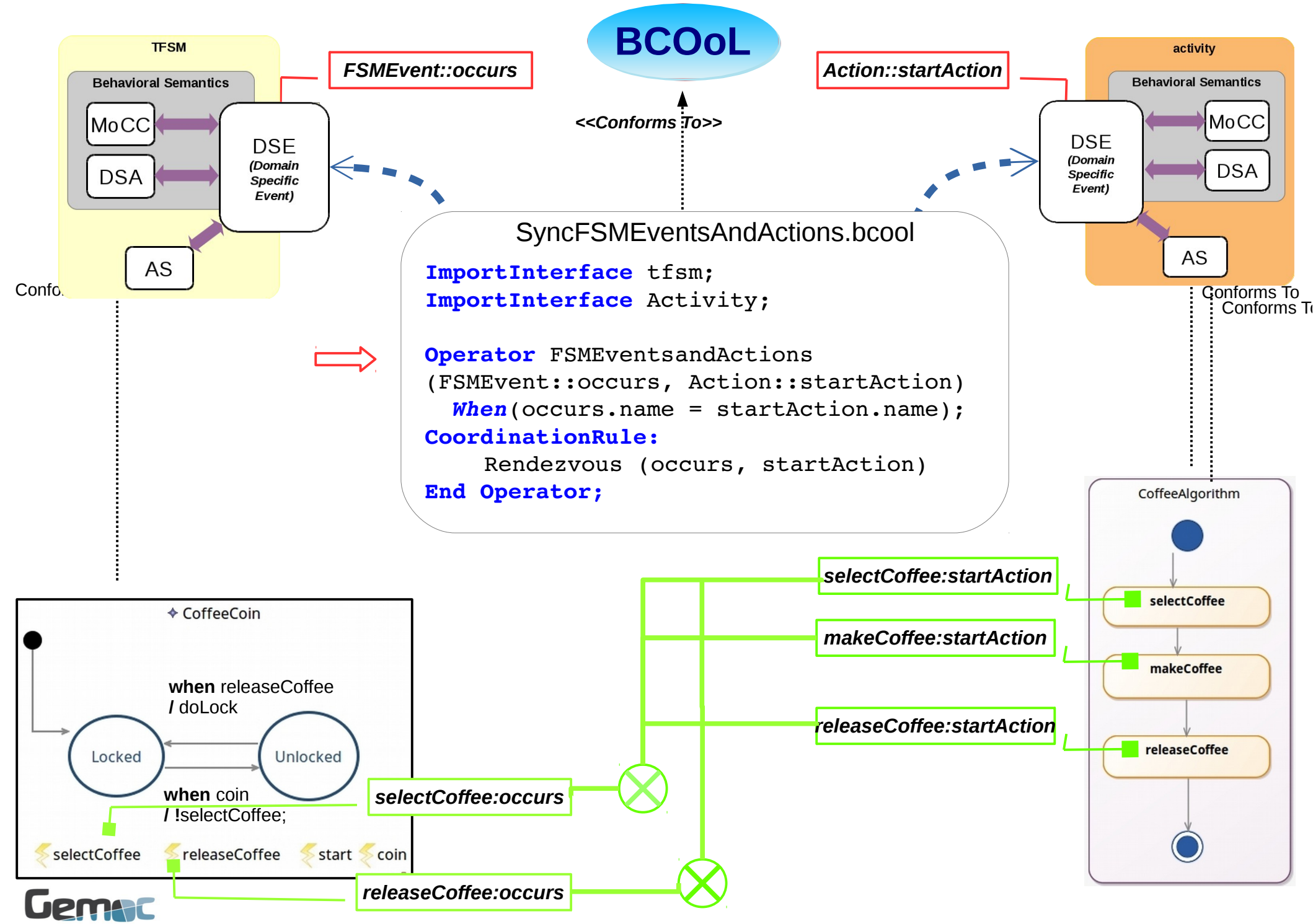
```
ImportInterface tfsm;
ImportInterface Activity;

Operator RendezVousWhenSameName
(FSMEvent::occurs,Action::startAction)

  When(occurs.name = startAction.name);

  CoordinationRule:
     RendezVous (occurs, startAction)
End Operator;
```

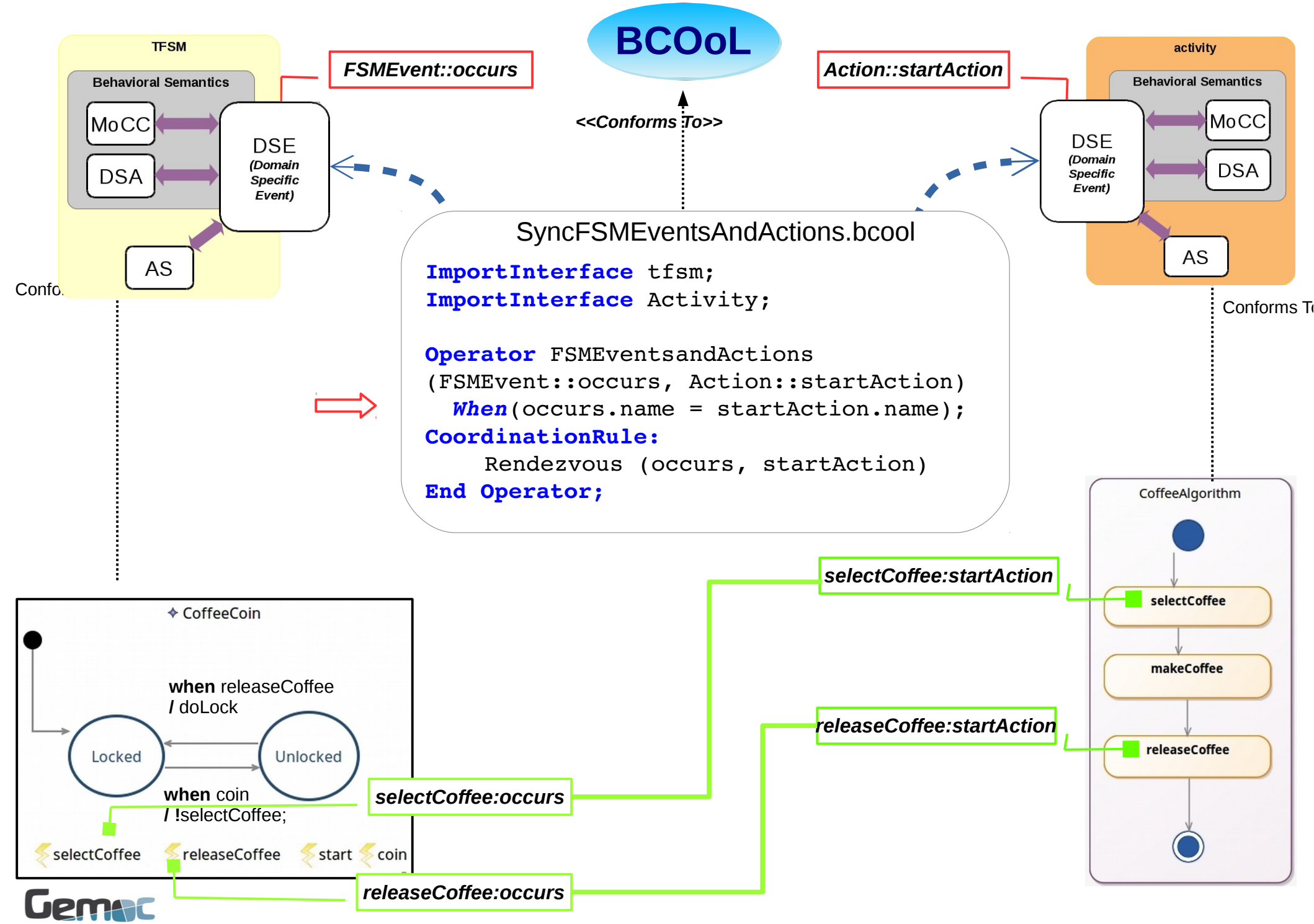
## Defined in MoCCML

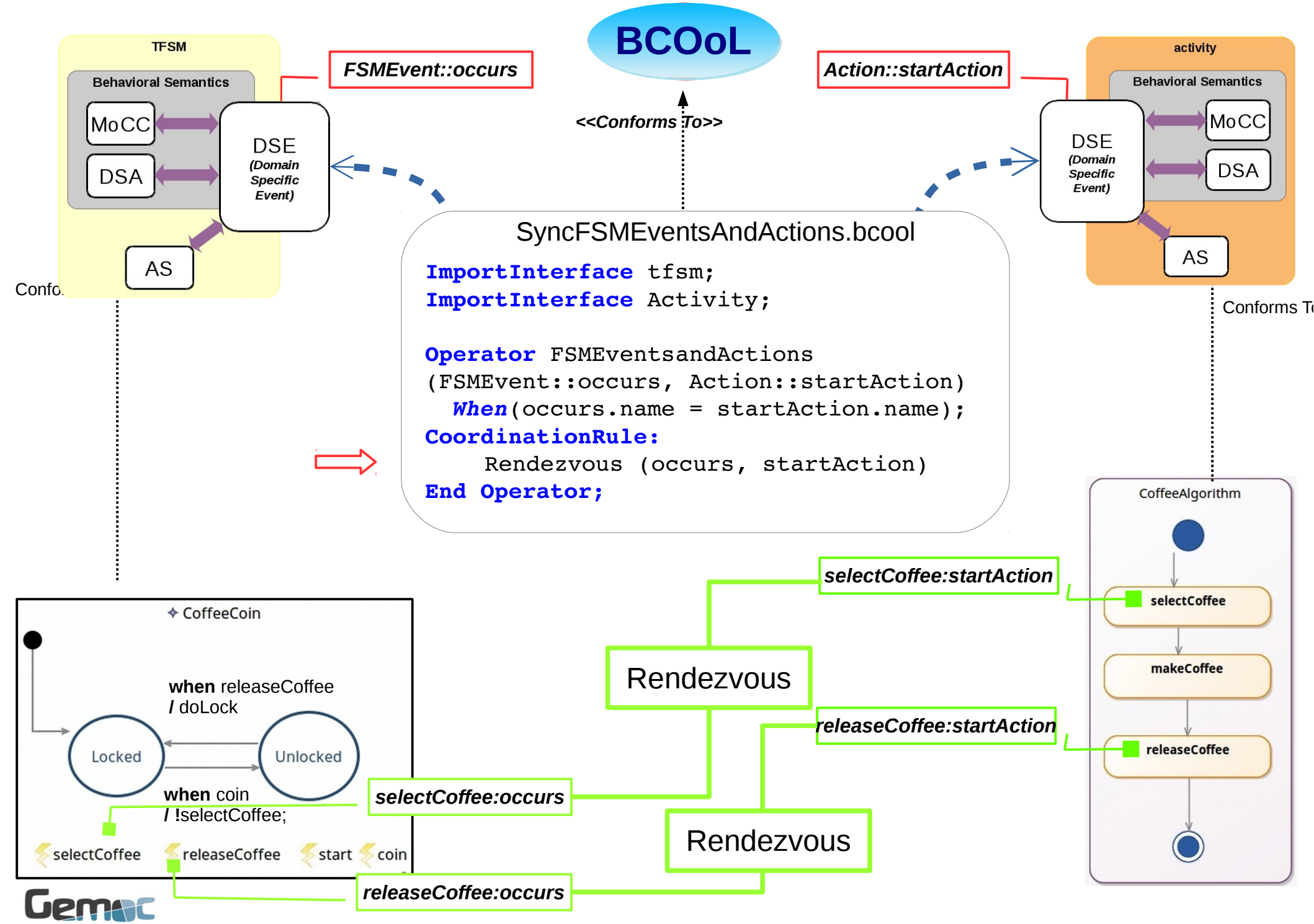
(Model of Concurrency and Communication Modeling Language)

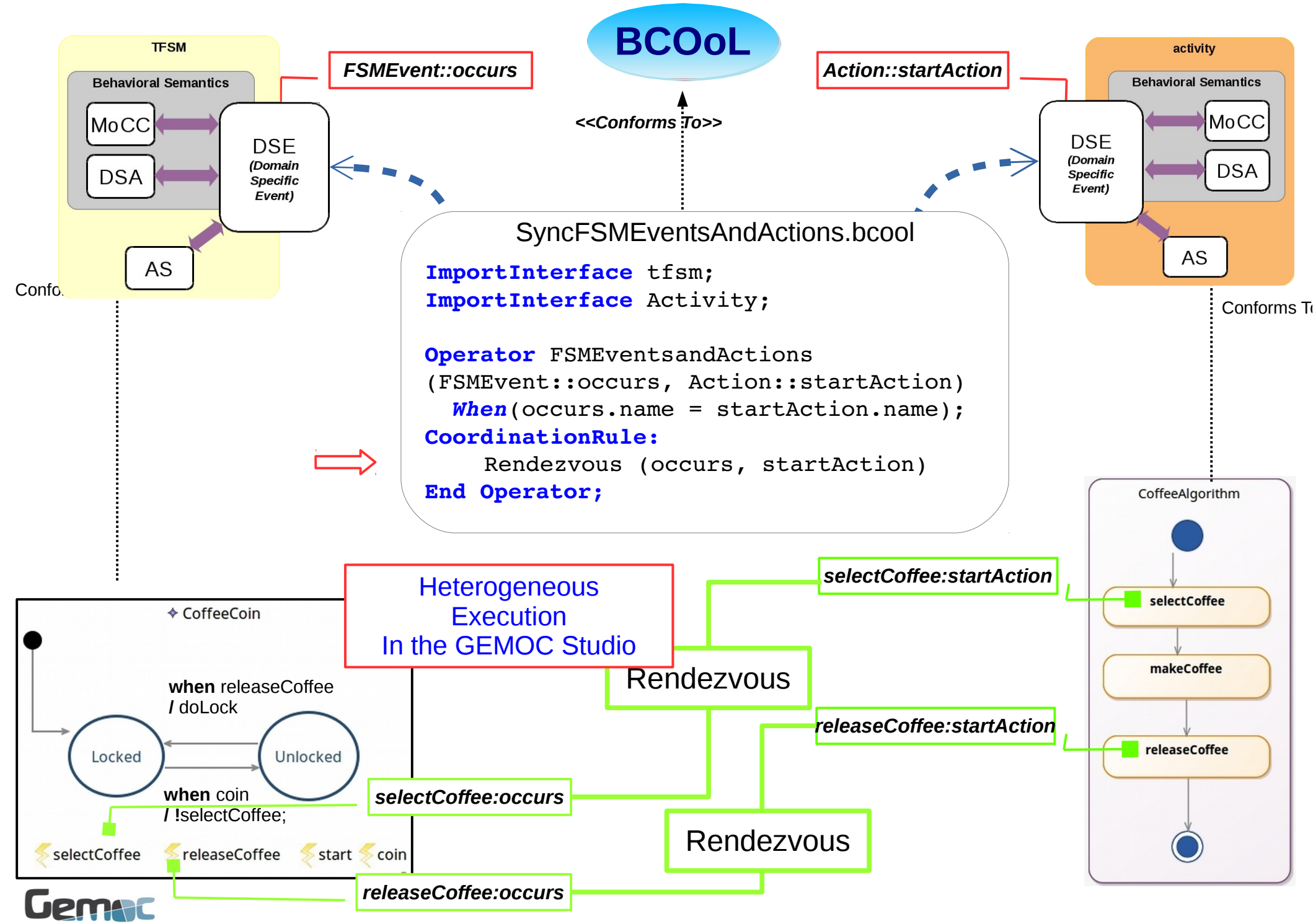

BCOoL
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
FSMEvent::occurs
<<Conforms To>>
Action::startAction
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
Conforms To
Conforms To
SyncFSMEventsAndActions.bcool
ImportInterface tfsm;
ImportInterface Activity;
Operator FSMEventsandActions
(FSMEvent::occurs, Action::startAction)
When(occurs.name = startAction.name);
CoordinationRule:
Rendezvous (occurs, startAction)
End Operator;
CoffeeCoin
when releaseCo
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
Locked:entering
Locked:leaving
Locked:entering
Locked:leaving
selectCoffee:occurs
releaseCoffee:occurs
selectCoffee:startAction
makeCoffee:startAction
makeCoffee:finishAction
releaseCoffee:startAction
releaseCoffee:finishAction
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee


BCOoL
<<Conforms To>>
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
FSMEvent::occurs
Action::startAction
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
Conforms To
Conforms To
SyncFSMEventsAndActions.bcool
ImportInterface tfsm;
ImportInterface Activity;
Operator FSMEventsandActions
(FSMEvent::occurs, Action::startAction)
When(occurs.name = startAction.name);
CoordinationRule:
Rendezvous (occurs, startAction)
End Operator;
CoffeeCoin
when releaseCoffee
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
selectCoffee:occurs
releaseCoffee:occurs
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee
selectCoffee:startAction
makeCoffee:startAction
releaseCoffee:startAction
Gemoc


BCOoL
<<Conforms To>>
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
FSMEvent::occurs
Action::startAction
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
Conforms To
Conforms To
SyncFSMEventsAndActions.bcool
ImportInterface tfsm;
ImportInterface Activity;
Operator FSMEventsandActions
(FSMEvent::occurs, Action::startAction)
When(occurs.name = startAction.name);
CoordinationRule:
Rendezvous (occurs, startAction)
End Operator;
CoffeeCoin
when releaseCoffee
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
selectCoffee:occurs
releaseCoffee:occurs
selectCoffee:startAction
makeCoffee:startAction
releaseCoffee:startAction
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee


BCOoL
<<Conforms To>>
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
FSMEvent::occurs
Action::startAction
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
Conforms To
SyncFSMEventsAndActions.bcool
ImportInterface tfsm;
ImportInterface Activity;
Operator FSMEventsandActions
(FSMEvent::occurs, Action::startAction)
When(occurs.name = startAction.name);
CoordinationRule:
Rendezvous (occurs, startAction)
End Operator;
CoffeeCoin
when releaseCoffee
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
selectCoffee:occurs
releaseCoffee:occurs
selectCoffee:startAction
releaseCoffee:startAction
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee


BCOoL
<<Conforms To>>
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
FSMEvent::occurs
Action::startAction
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
Conforms To
SyncFSMEventsAndActions.bcool
ImportInterface tfsm;
ImportInterface Activity;
Operator FSMEventsandActions
(FSMEvent::occurs, Action::startAction)
When(occurs.name = startAction.name);
CoordinationRule:
Rendezvous (occurs, startAction)
End Operator;
CoffeeCoin
when releaseCoffee
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
selectCoffee:occurs
releaseCoffee:occurs
Rendezvous
Rendezvous
selectCoffee:startAction
releaseCoffee:startAction
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee
Gemoc


BCOoL
<<Conforms To>>
TFSM
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
FSMEvent::occurs
Action::startAction
activity
Behavioral Semantics
MoCC
DSA
DSE
(Domain Specific Event)
AS
Conforms To
SyncFSMEventsAndActions.bcool
ImportInterface tfsm;
ImportInterface Activity;
Operator FSMEventsandActions
(FSMEvent::occurs, Action::startAction)
When(occurs.name = startAction.name);
CoordinationRule:
Rendezvous (occurs, startAction)
End Operator;
Heterogeneous Execution In the GEMOC Studio
CoffeeCoin
when releaseCoffee
/ doLock
Locked
Unlocked
when coin
/ !selectCoffee;
selectCoffee
releaseCoffee
start
coin
selectCoffee:occurs
releaseCoffee:occurs
Rendezvous
Rendezvous
selectCoffee:startAction
releaseCoffee:startAction
CoffeeAlgorithm
selectCoffee
makeCoffee
releaseCoffee
Gemoc

# Implemented into the GEMOC studio

syncproducttfsmwithfuml.bcool

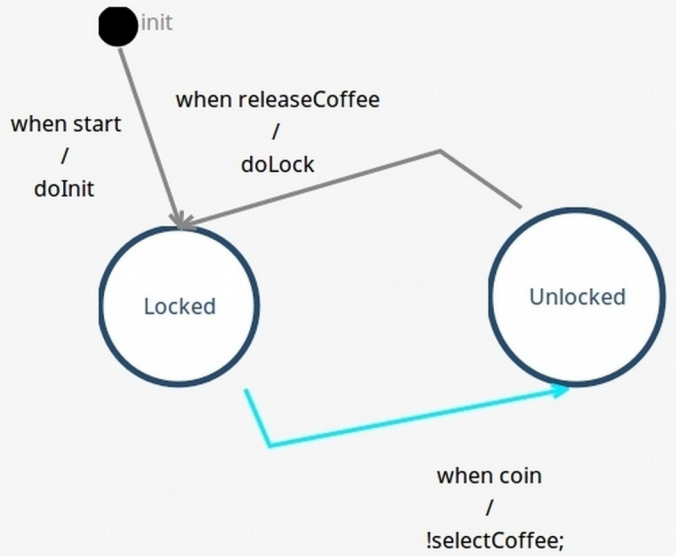
```
SyncProductTfsmwithfUML

ImportLib "platform:/resource/org.gemoc.bcool.example.productfumlandtfsm/operator/facilities.bcoollib"
ImportLib "platform:/resource/org.gemoc.bcool.example.productfumlandtfsm/operator/bcoollib.ccslLib"

ImportInterface "platform:/plugin/org.modelexecution.operationalsemantics.gemoc.ecl/model/ActivitiyDiagramV2.ecl" as ad
ImportInterface "platform:/plugin/org.gemoc.sample.tfsm.eclmoc2as/ecl/TFSM.ecl" as tfsm

Spec test

Operator  MatchingandCoordinationSharedEvents (dse1 : tfsm::occurs, dse2 : ad::executeIt)
MatchingCorrespondance: when  dse1.name = dse2.name ;
       CoordinationRule:
          facilities.RendezVous(dse1, dse2)
end operator;
```

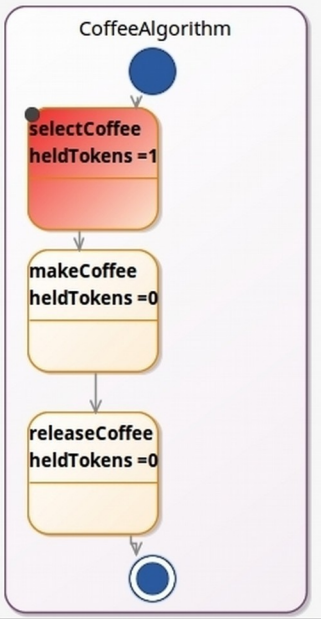
Editing facilities by using Xtext

Concurrent Logical Steps Decider

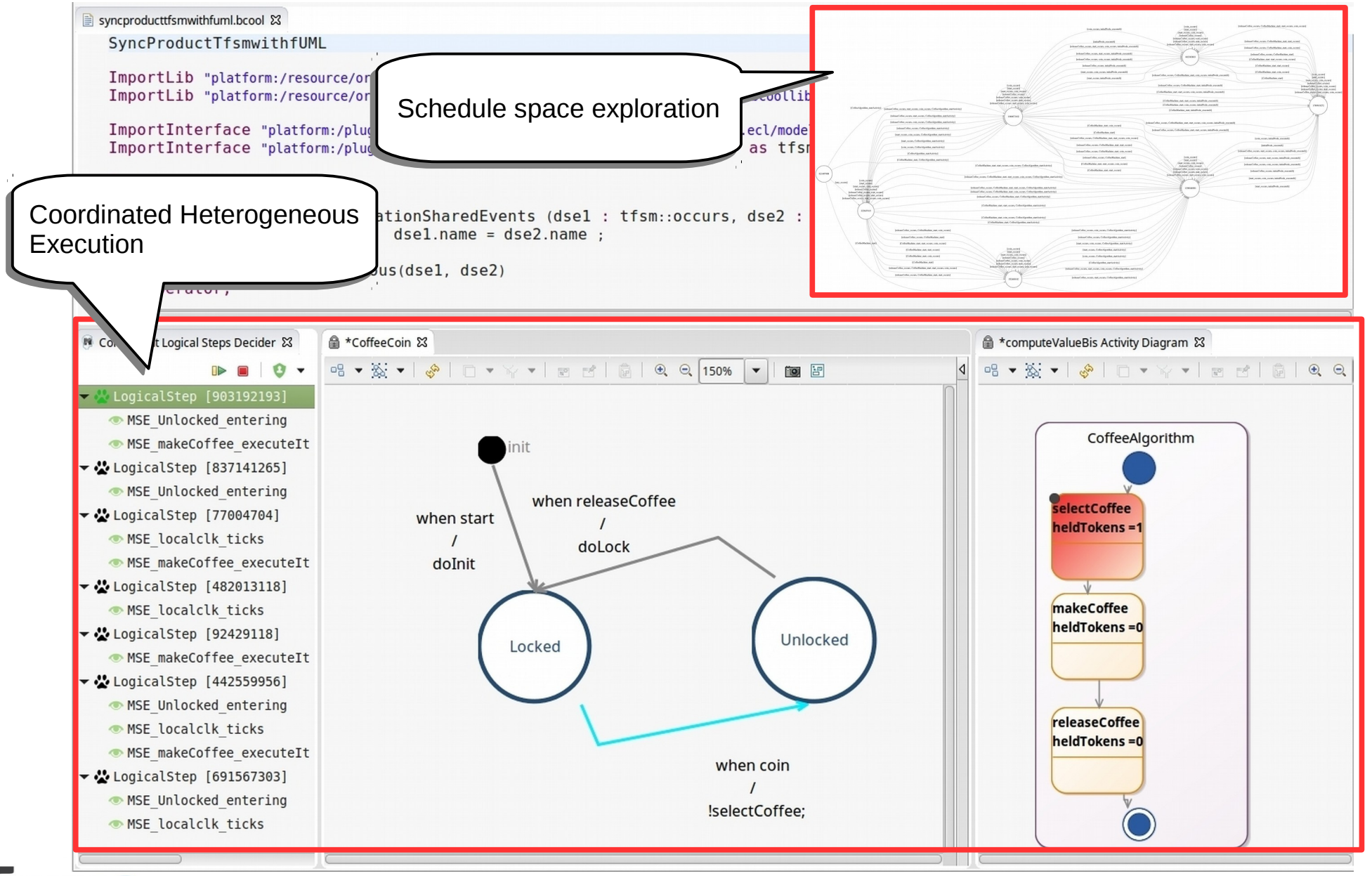
- LogicalStep [903192193]
  - MSE_Unlocked_entering
  - MSE_makeCoffee_executeIt
- LogicalStep [837141265]
  - MSE_Unlocked_entering
- LogicalStep [77004704]
  - MSE_localclk_ticks
  - MSE_makeCoffee_executeIt
- LogicalStep [482013118]
  - MSE_localclk_ticks
- LogicalStep [92429118]
  - MSE_makeCoffee_executeIt
- LogicalStep [442559956]
  - MSE_Unlocked_entering
  - MSE_localclk_ticks
  - MSE_makeCoffee_executeIt
- LogicalStep [691567303]
  - MSE_Unlocked_entering
  - MSE_localclk_ticks

*CoffeeCoin



# Implemented into the GEMOC studio



# Conclusion

- BCOoL is a dedicated metalanguage to capture coordination patterns.
- It automates the coordination of models by relying on a formal language.
- It is associated to the GEMOC language/modeling workbench to execute and analyze the coordinated system.

➔ Future work:

  - Using the explicit coordination to generate master on co-simulation bus
  - Understanding the interconnection with physical model (continuous time)

[1]http://timesquare.inria.fr/BCOoL

# Thanks

http://timesquare.inria.fr/BCOoL

http://gemoc.org/ins

BCOoL

Gemoc