Experimentations, transfer and development

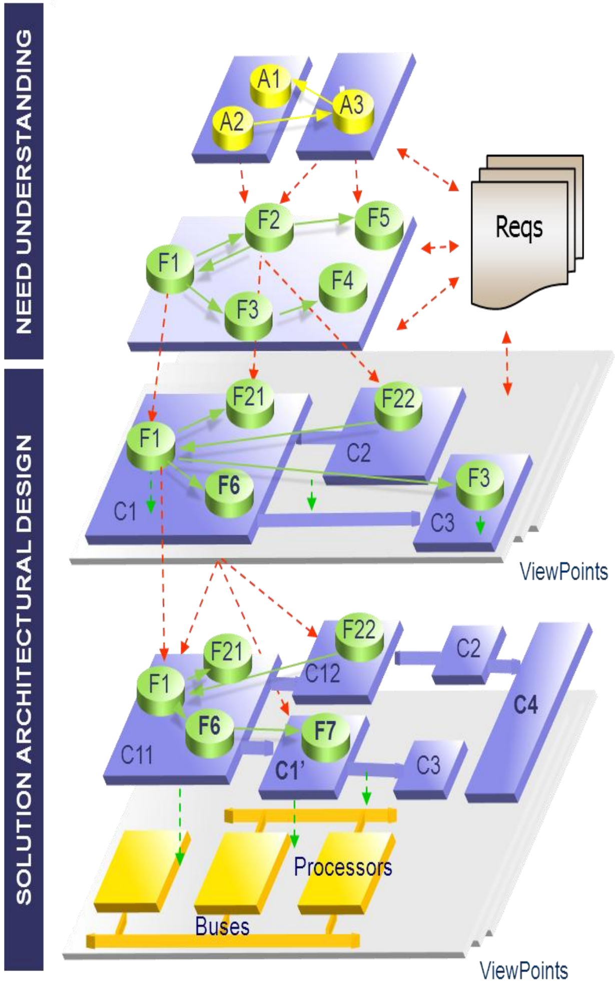
Final workshop of the ANR project GEMOC, March 17th, 2016

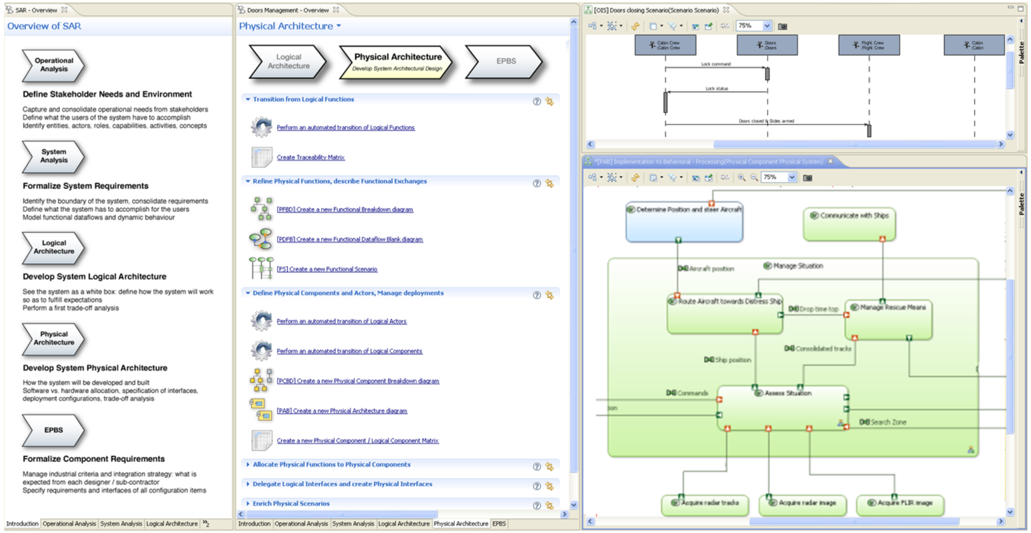
Jérôme Le Noir (Thales Research & Technology)

ANR-12-INSE-0011

Industrial Context

ARCADIA




SAR - Overview

Overview of SAR

Operational Analysis

Define Stakeholder Needs and Environment

Capture and consolidate operational needs from stakeholders
Define what the users of the system have to accomplish
Identify entities, actors, roles, capabilities, activities, concepts

System Analysis

Formalize System Requirements

Identify the boundary of the system, consolidate requirements
Define what the system has to accomplish for the users
Model functional dataflows and dynamic behaviour

Logical Architecture

Develop System Logical Architecture

See the system as a white box: define how the system will work so as to fulfill expectations
Perform a first trade-off analysis

Physical Architecture

Develop System Physical Architecture

How the system will be developed and built
Software vs. hardware allocation, specification of interfaces, deployment configurations, trade-off analysis

EPBS

Formalize Component Requirements

Manage industrial criteria and integration strategy: what is expected from each designer / sub-contractor
Specify requirements and interfaces of all configuration items

Introduction | Operational Analysis | System Analysis | Logical Architecture

Doors Management - Overview

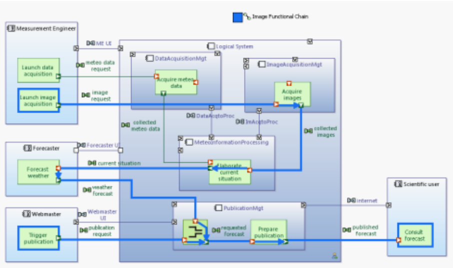
Physical Architecture

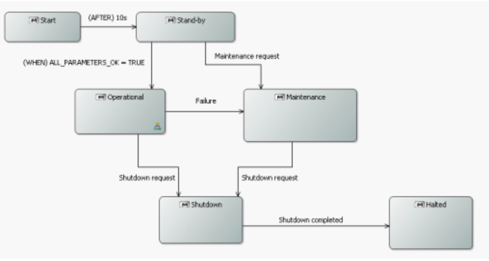
Logical Architecture | Physical Architecture *Develop System Architectural Design* | EPBS

- Transition from Logical Functions
  - Perform an automated transition of Logical Functions
  - Create Traceability Matrix
- Refine Physical Functions, describe Functional Exchanges
  - [PFBD] Create a new Functional Breakdown diagram
  - [PDFB] Create a new Functional Dataflow Blank diagram
  - [FS] Create a new Functional Scenario
- Define Physical Components and Actors, Manage deployments
  - Perform an automated transition of Logical Actors
  - Perform an automated transition of Logical Components
  - [PCBD] Create a new Physical Component Breakdown diagram
  - [PAB] Create a new Physical Architecture diagram
  - Create a new Physical Component / Logical Component Matrix
- Allocate Physical Functions to Physical Components
- Delegate Logical Interfaces and create Physical Interfaces
- Enrich Physical Scenarios

Introduction | Operational Analysis | System Analysis | Logical Architecture | Physical Architecture | EPBS

Capella

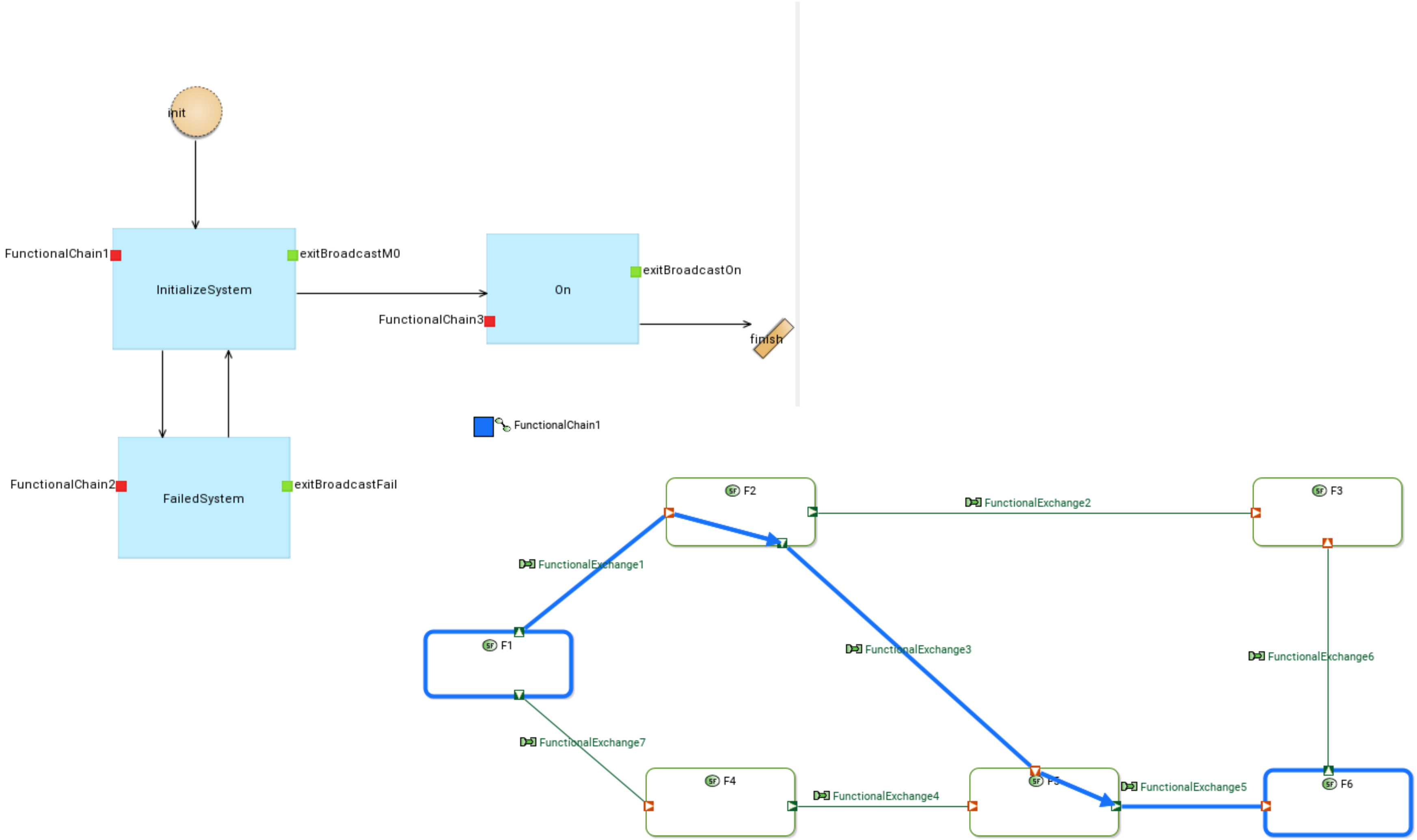




https://www.polarsys.org/capella/arcadia.html

Can we coordinate this heterogeneous model ?

Mode Automata & DataFlow Model Coordination

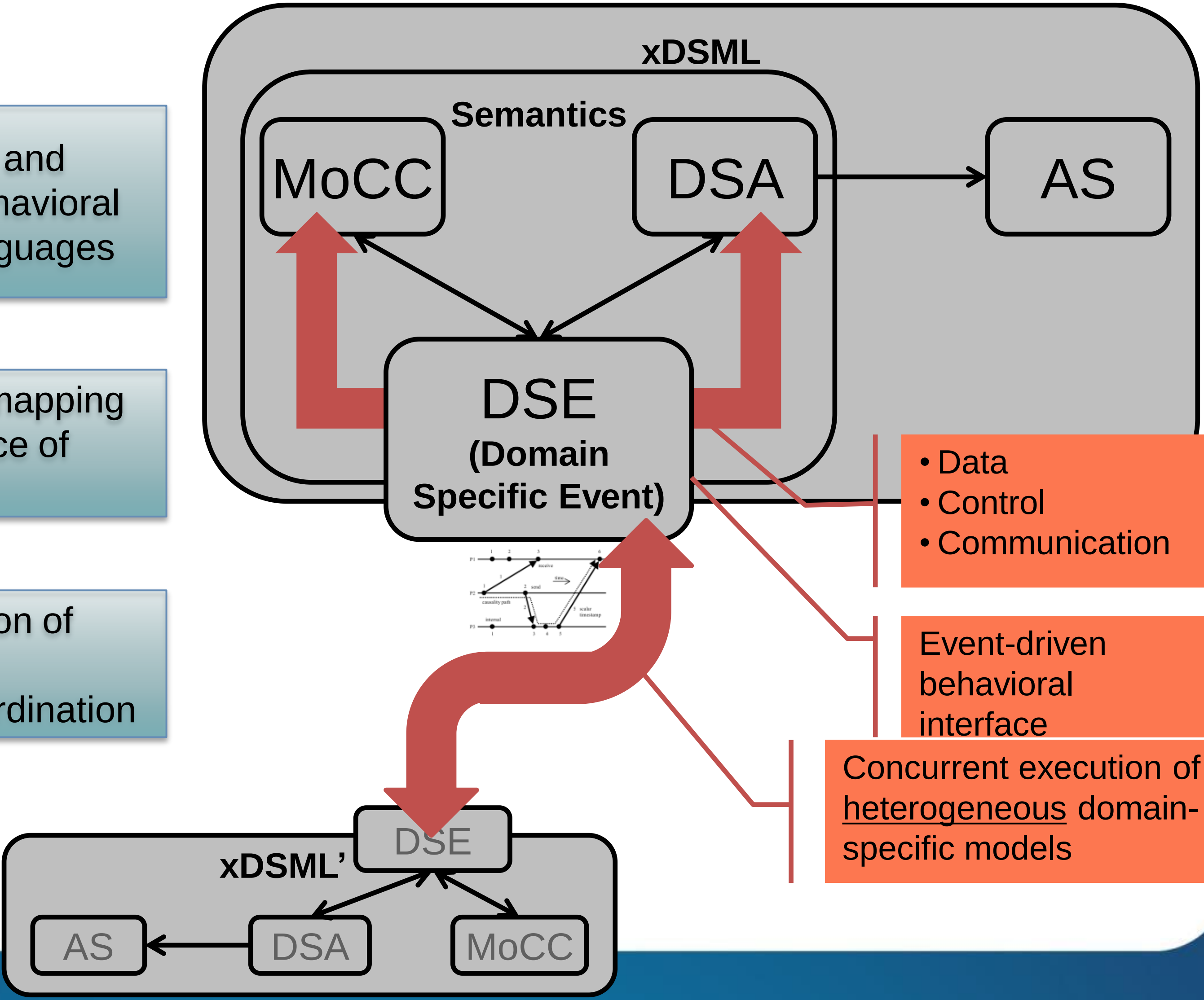


xDSML Development and Composition

Breakthrough #1: modular and explicit definition of the behavioral semantics of modeling languages

Breakthrough #2: explicit mapping used as behavioral interface of modeling languages

Breakthrough #3: integration of modeling languages for heterogeneous model coordination



- Data
- Control
- Communication

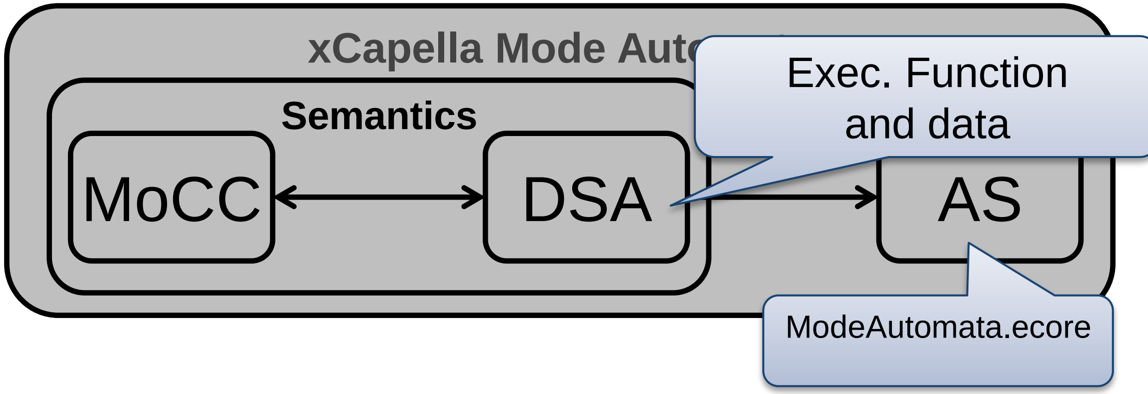
Event-driven behavioral interface

Concurrent execution of heterogeneous domain-specific models

ANR-12-INSE-0011 ANR Gemoc

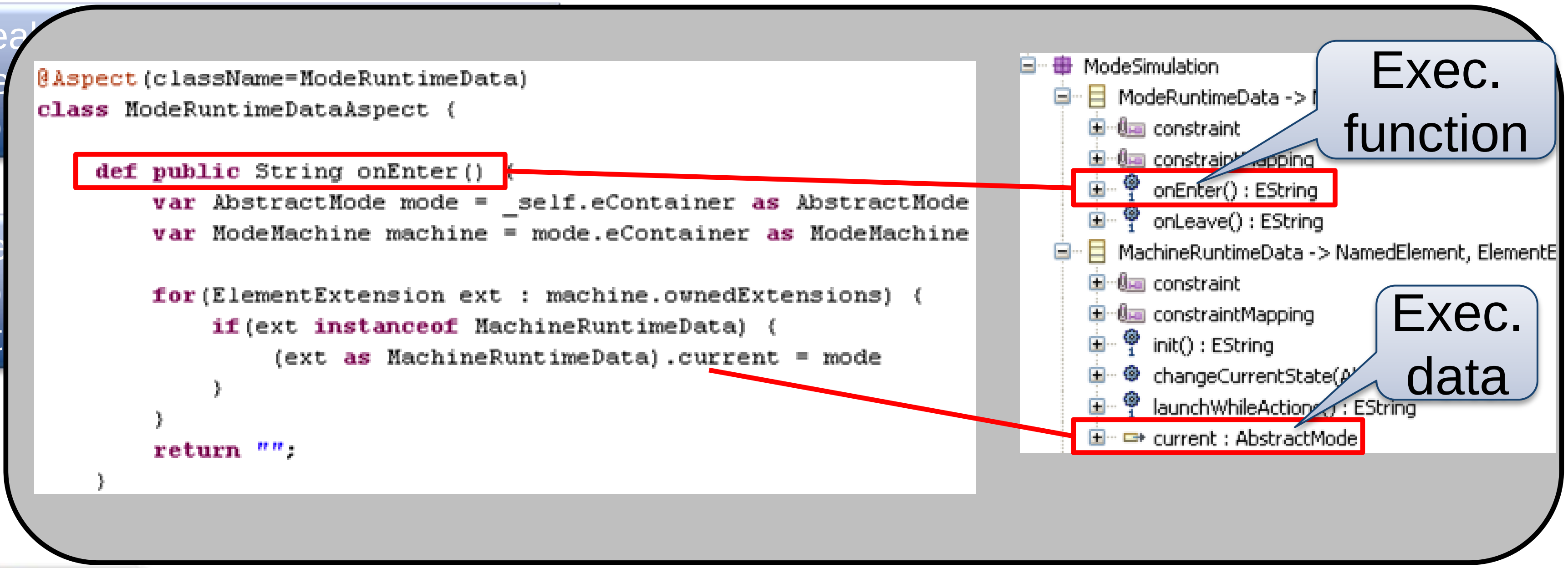
xCapella Mode Automata : DSA

Breakthrough #1: modular and explicit definition of the behavioral semantics of modeling languages



Brea
use
mo

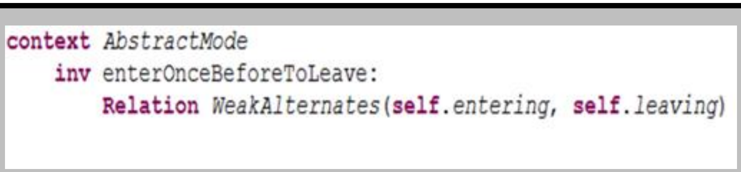
Bre
mo
het



```
@Aspect(className=ModeRuntimeData)
class ModeRuntimeDataAspect {

   def public String onEnter() {
       var AbstractMode mode = _self.eContainer as AbstractMode
       var ModeMachine machine = mode.eContainer as ModeMachine

       for(ElementExtension ext : machine.ownedExtensions) {
           if(ext instanceof MachineRuntimeData) {
               (ext as MachineRuntimeData).current = mode
           }
       }
       return "";
   }
```

```
ModeSimulation
   ModeRuntimeData -> I
      constraint
      constrai   apping
      onEnter() : EString
      onLeave() : EString
   MachineRuntimeData -> NamedElement, ElementE
      constraint
      constraintMapping
      init() : EString
      changeCurrentState(A
      launchWhileAction( ) : EString
      current : AbstractMode
```

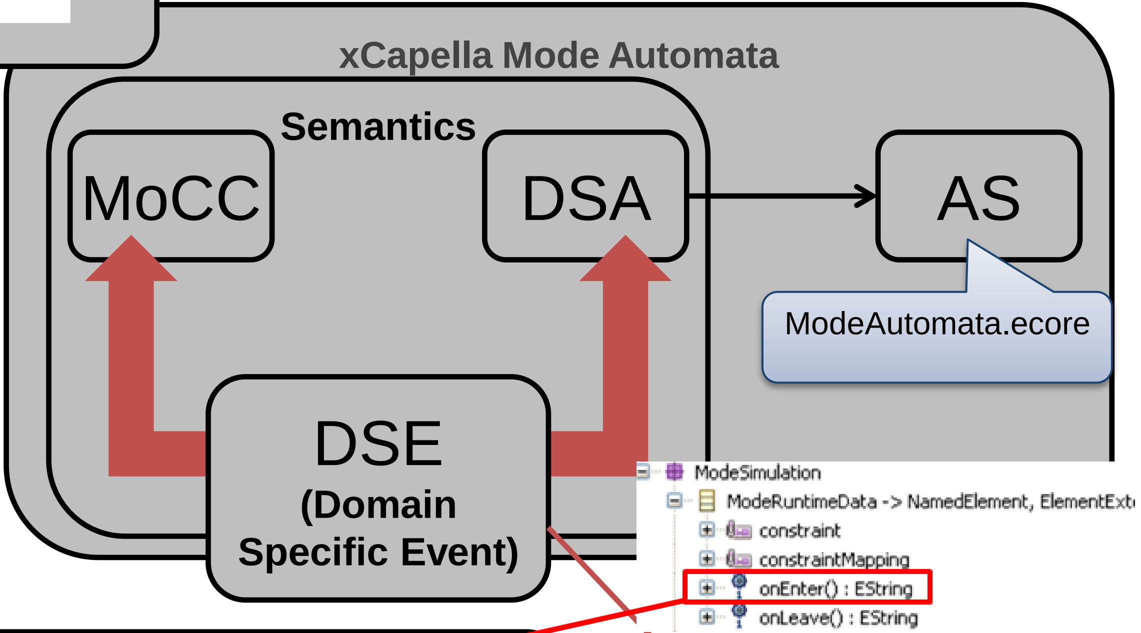
xCapella Mode Automata : DSE & MoCC



```
context AbstractMode
    inv enterOnceBeforeToLeave:
        Relation WeakAlternates(self.entering, self.leaving)
```

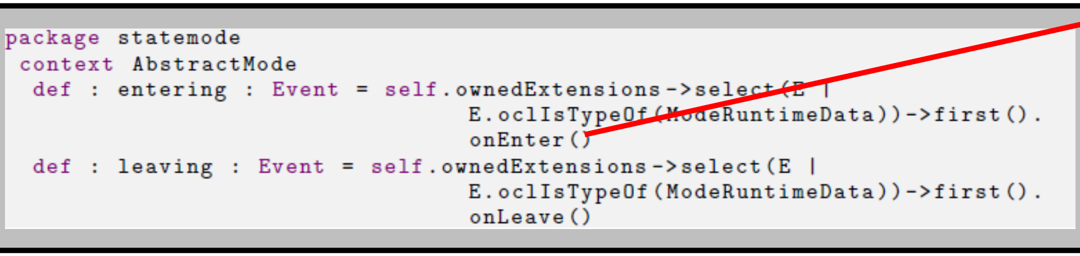
Breakthrough #1: modular and explicit definition of the behavioral semantics of modeling languages ✓

Breakthrough #2: explicit mapping used as behavioral interface of modeling languages ✓



```
ModeSimulation
  ModeRuntimeData -> NamedElement, ElementExt
    constraint
    constraintMapping
    onEnter() : EString
    onLeave() : EString
```

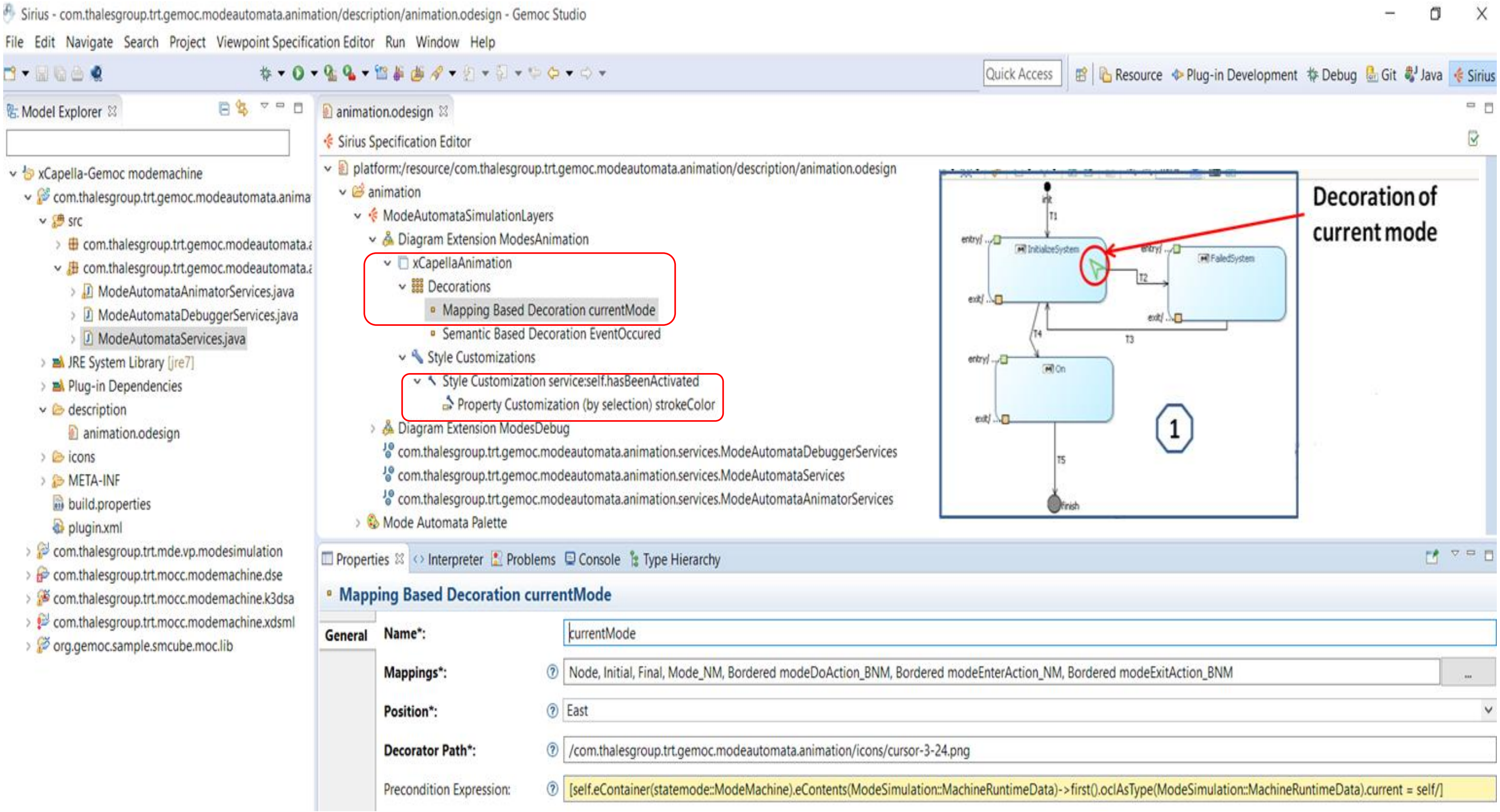
Event-driven behavioral interface



```
package statemode
 context AbstractMode
  def : entering : Event = self.ownedExtensions->select(E |
                              E.oclIsTypeOf(ModeRuntimeData))->first().
                              onEnter()
  def : leaving : Event = self.ownedExtensions->select(E |
                              E.oclIsTypeOf(ModeRuntimeData))->first().
                              onLeave()
```

ANR-12-INSE-0011 ANR

Mode automata Animator



Sirius - com.thalesgroup.trt.gemoc.modeautomata.animation/description/animation.odesign - Gemoc Studio
File Edit Navigate Search Project Viewpoint Specification Editor Run Window Help
Quick Access
Resource
Plug-in Development
Debug
Git
Java
Sirius
Model Explorer
animation.odesign
Sirius Specification Editor

- xCapella-Gemoc modemachine
  - com.thalesgroup.trt.gemoc.modeautomata.anima
    - src
      - com.thalesgroup.trt.gemoc.modeautomata.a
      - com.thalesgroup.trt.gemoc.modeautomata.a
        - ModeAutomataAnimatorServices.java
        - ModeAutomataDebuggerServices.java
        - ModeAutomataServices.java
    - JRE System Library [jre7]
    - Plug-in Dependencies
    - description
      - animation.odesign
    - icons
    - META-INF
    - build.properties
    - plugin.xml
- com.thalesgroup.trt.mde.vp.modesimulation
- com.thalesgroup.trt.mocc.modemachine.dse
- com.thalesgroup.trt.mocc.modemachine.k3dsa
- com.thalesgroup.trt.mocc.modemachine.xdsml
- org.gemoc.sample.smcube.moc.lib

- platform:/resource/com.thalesgroup.trt.gemoc.modeautomata.animation/description/animation.odesign
  - animation
    - ModeAutomataSimulationLayers
      - Diagram Extension ModesAnimation
        - xCapellaAnimation
          - Decorations
            - Mapping Based Decoration currentMode
            - Semantic Based Decoration EventOccured
          - Style Customizations
            - Style Customization service:self.hasBeenActivated
              - Property Customization (by selection) strokeColor
      - Diagram Extension ModesDebug
      - com.thalesgroup.trt.gemoc.modeautomata.animation.services.ModeAutomataDebuggerServices
      - com.thalesgroup.trt.gemoc.modeautomata.animation.services.ModeAutomataServices
      - com.thalesgroup.trt.gemoc.modeautomata.animation.services.ModeAutomataAnimatorServices
    - Mode Automata Palette

Decoration of current mode

Mapping Based Decoration currentMode

General

Name*: currentMode

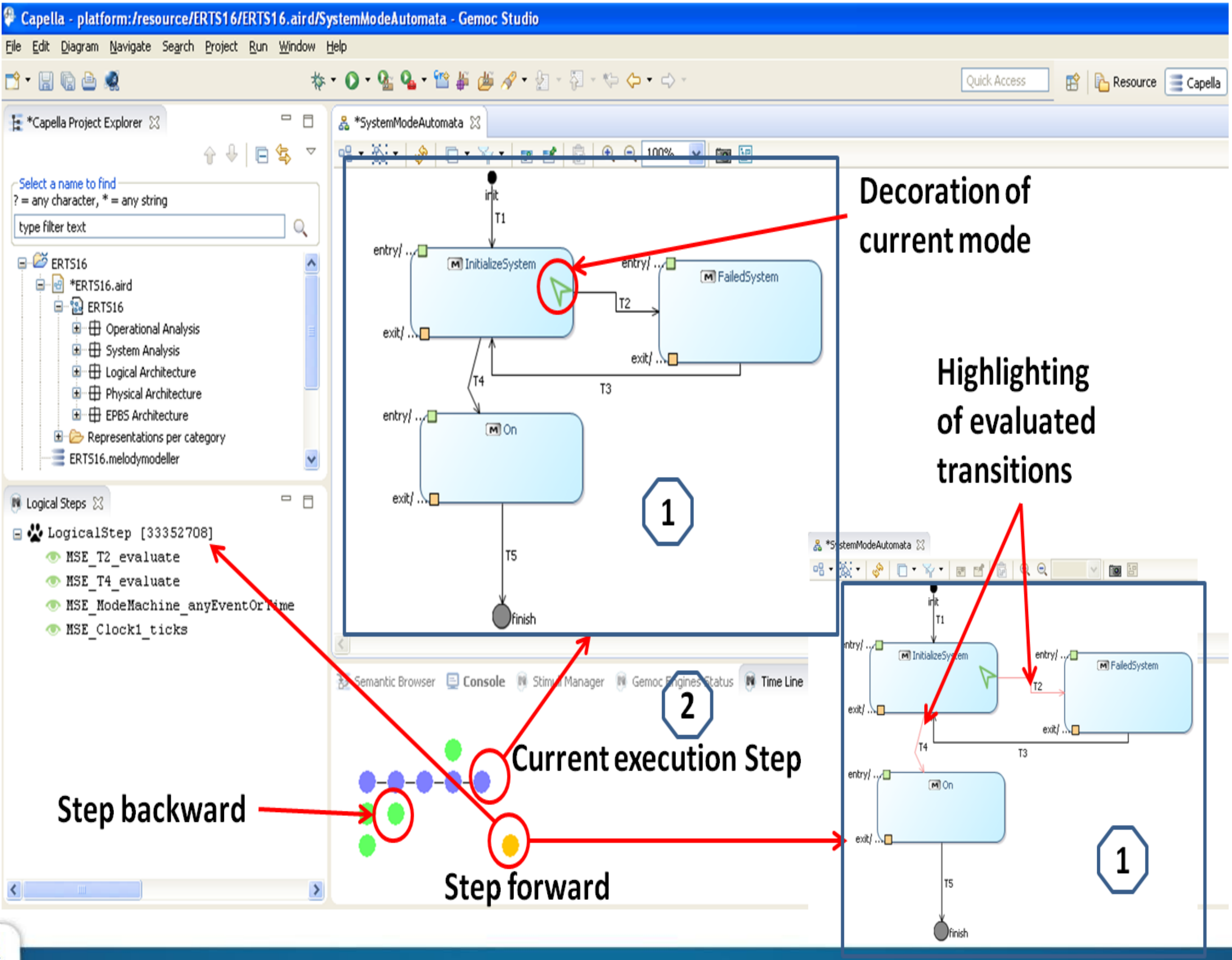
Mappings*: Node, Initial, Final, Mode_NM, Bordered modeDoAction_BNM, Bordered modeEnterAction_NM, Bordered modeExitAction_BNM

Position*: East

Decorator Path*: /com.thalesgroup.trt.gemoc.modeautomata.animation/icons/cursor-3-24.png

Precondition Expression: [self.eContainer(statemode::ModeMachine).eContents(ModeSimulation::MachineRuntimeData)->first().oclAsType(ModeSimulation::MachineRuntimeData).current = self/]

Breath life into an industrial modeling workbench with the GEMOC approach



Decoration of current mode

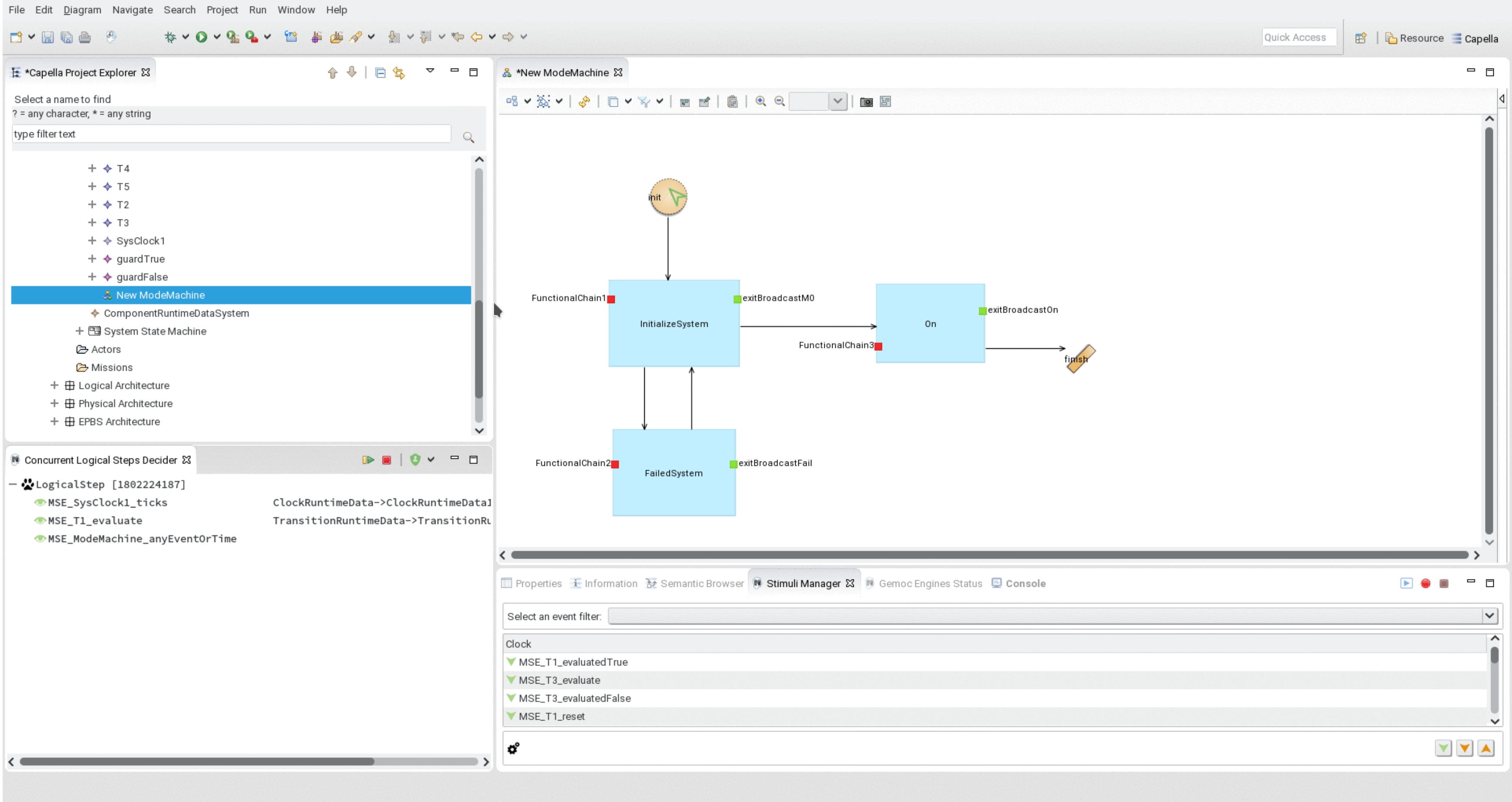
Highlighting of evaluated transitions

Current execution Step

Step backward

Step forward

xCapella Mode Automata



Select a name to find
? = any character, * = any string

type filter text

- T4
- T5
- T2
- T3
- SysClock1
- guardTrue
- guardFalse
- New ModeMachine
- ComponentRuntimeDataSystem
- System State Machine
- Actors
- Missions
- Logical Architecture
- Physical Architecture
- EPBS Architecture

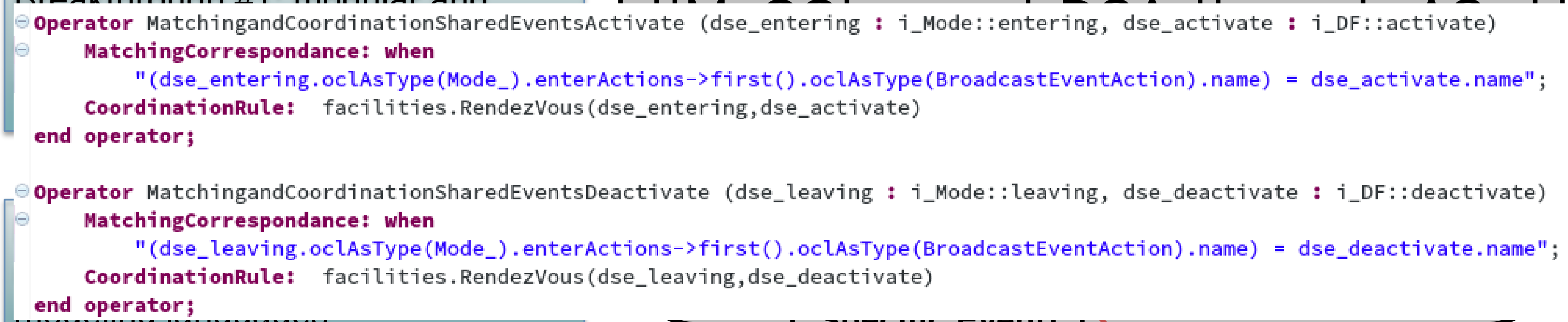
```
LogicalStep [1802224187]
  MSE_SysClock1_ticks                 ClockRuntimeData->ClockRuntimeDataI
  MSE_T1_evaluate                     TransitionRuntimeData->TransitionRu
  MSE_ModeMachine_anyEventOrTime
```

Select an event filter:

Clock
MSE_T1_evaluatedTrue
MSE_T3_evaluate
MSE_T3_evaluatedFalse
MSE_T1_reset

xCapella Mode Automata & DataFlow Coordination Specification

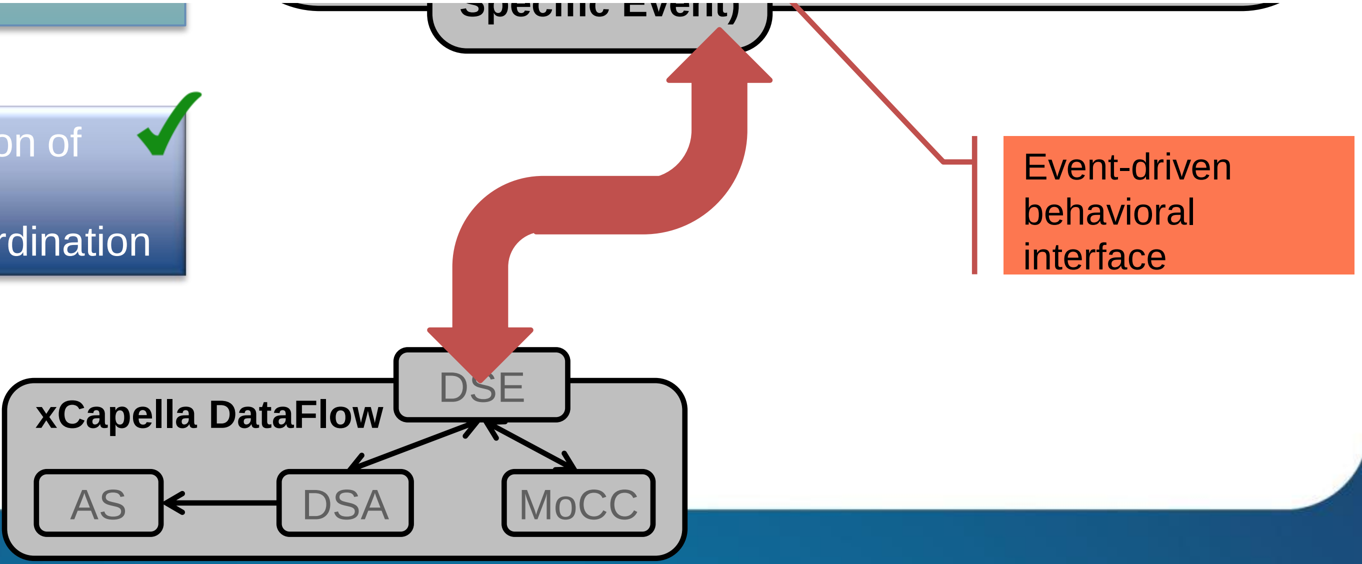
Breakthrough #1: modular and



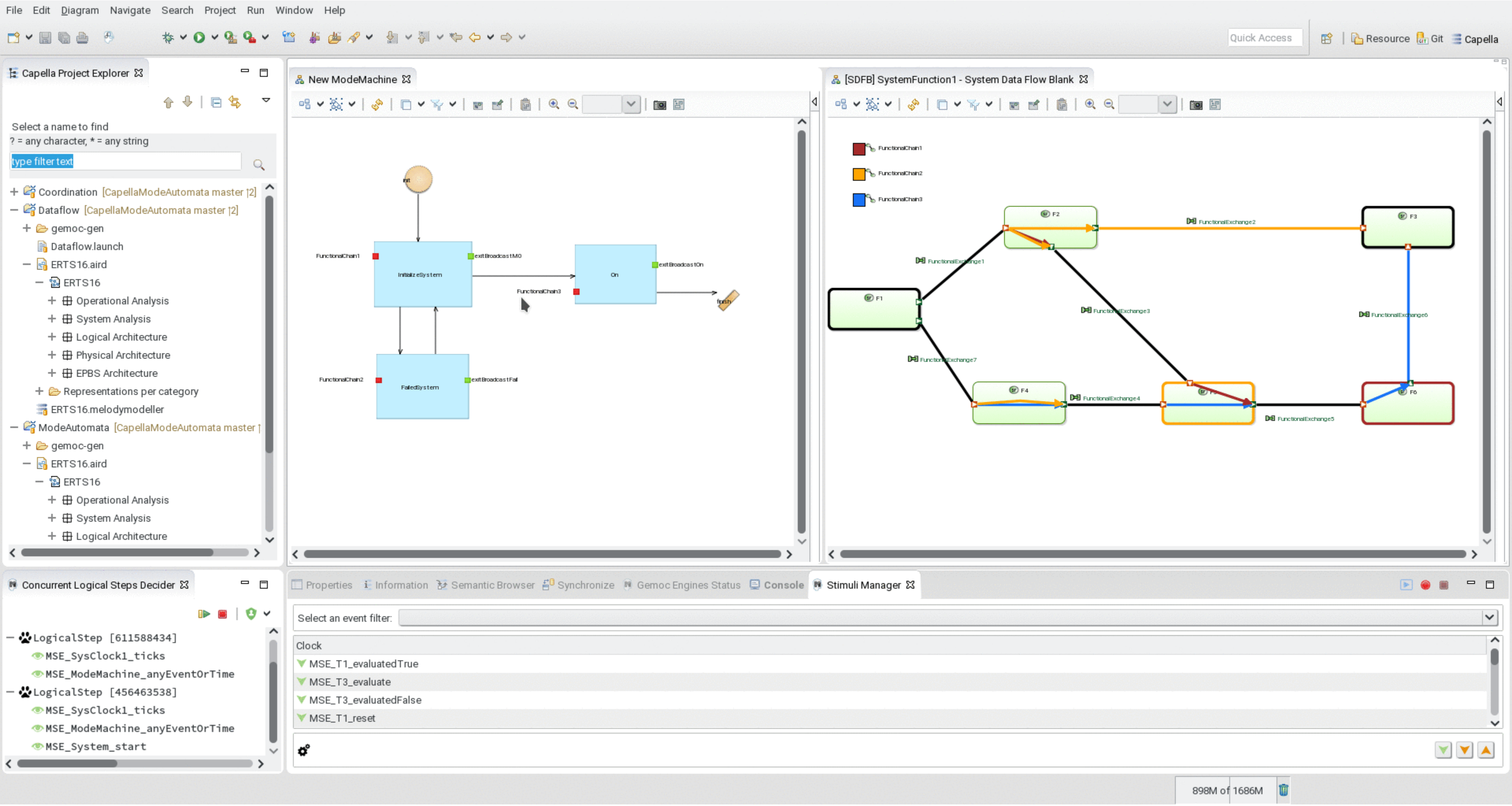
```
Operator MatchingandCoordinationSharedEventsActivate (dse_entering : i_Mode::entering, dse_activate : i_DF::activate)
    MatchingCorrespondance: when
        "(dse_entering.oclAsType(Mode_).enterActions->first().oclAsType(BroadcastEventAction).name) = dse_activate.name";
    CoordinationRule:  facilities.RendezVous(dse_entering,dse_activate)
end operator;

Operator MatchingandCoordinationSharedEventsDeactivate (dse_leaving : i_Mode::leaving, dse_deactivate : i_DF::deactivate)
    MatchingCorrespondance: when
        "(dse_leaving.oclAsType(Mode_).enterActions->first().oclAsType(BroadcastEventAction).name) = dse_deactivate.name";
    CoordinationRule:  facilities.RendezVous(dse_leaving,dse_deactivate)
end operator;
```

modeling languages

Breakthrough #3: integration of modeling languages for heterogeneous model coordination ✓



Mode Automata & Data Flow Model Coordination



Select a name to find
? = any character, * = any string
type filter text

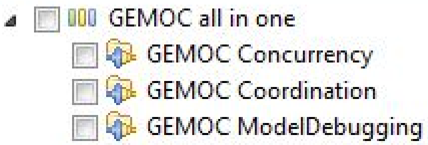
- Coordination [CapellaModeAutomata master ↑2]
- Dataflow [CapellaModeAutomata master ↑2]
  - gemoc-gen
  - Dataflow.launch
  - ERTS16.aird
    - ERTS16
      - Operational Analysis
      - System Analysis
      - Logical Architecture
      - Physical Architecture
      - EPBS Architecture
    - Representations per category
  - ERTS16.melodymodeller
- ModeAutomata [CapellaModeAutomata master ↑
  - gemoc-gen
  - ERTS16.aird
    - ERTS16
      - Operational Analysis
      - System Analysis
      - Logical Architecture

```
LogicalStep [611588434]
  MSE_SysClock1_ticks
  MSE_ModeMachine_anyEventOrTime
LogicalStep [456463538]
  MSE_SysClock1_ticks
  MSE_ModeMachine_anyEventOrTime
  MSE_System_start
```

Select an event filter:

Clock
MSE_T1_evaluatedTrue
MSE_T3_evaluate
MSE_T3_evaluatedFalse
MSE_T1_reset

Transfer : GEMOC Studio



- □ GEMOC all in one
  - □ GEMOC Concurrency
  - □ GEMOC Coordination
  - □ GEMOC ModelDebugging

- GEMOC studio : http://gemoc.org/studio-download
- GEMOC studio update site: http://gemoc.org/updatesite/studio

1. Model debuging: (sequential) execution, trace management and animation
   - Requires: Xtend/Java, generative approach for trace management, generic execution engine, generic animation framework

2. Concurrency modeling and analysis: (concurrent) execution, and analysis tools
   - Requires: model debugging + MoccML, ECL/GEL, Timesquare, concurrent execution engine

3. Behavioral coordination of, possibly heterogeneous, models: coordination engine
   - Requires: concurrency + BCOoL, coordination execution engine

Transfer : Experimentations

GEMOC Studio examples (deployed) :

- TFSM
- SigPML

Public GEMOC experimentations :

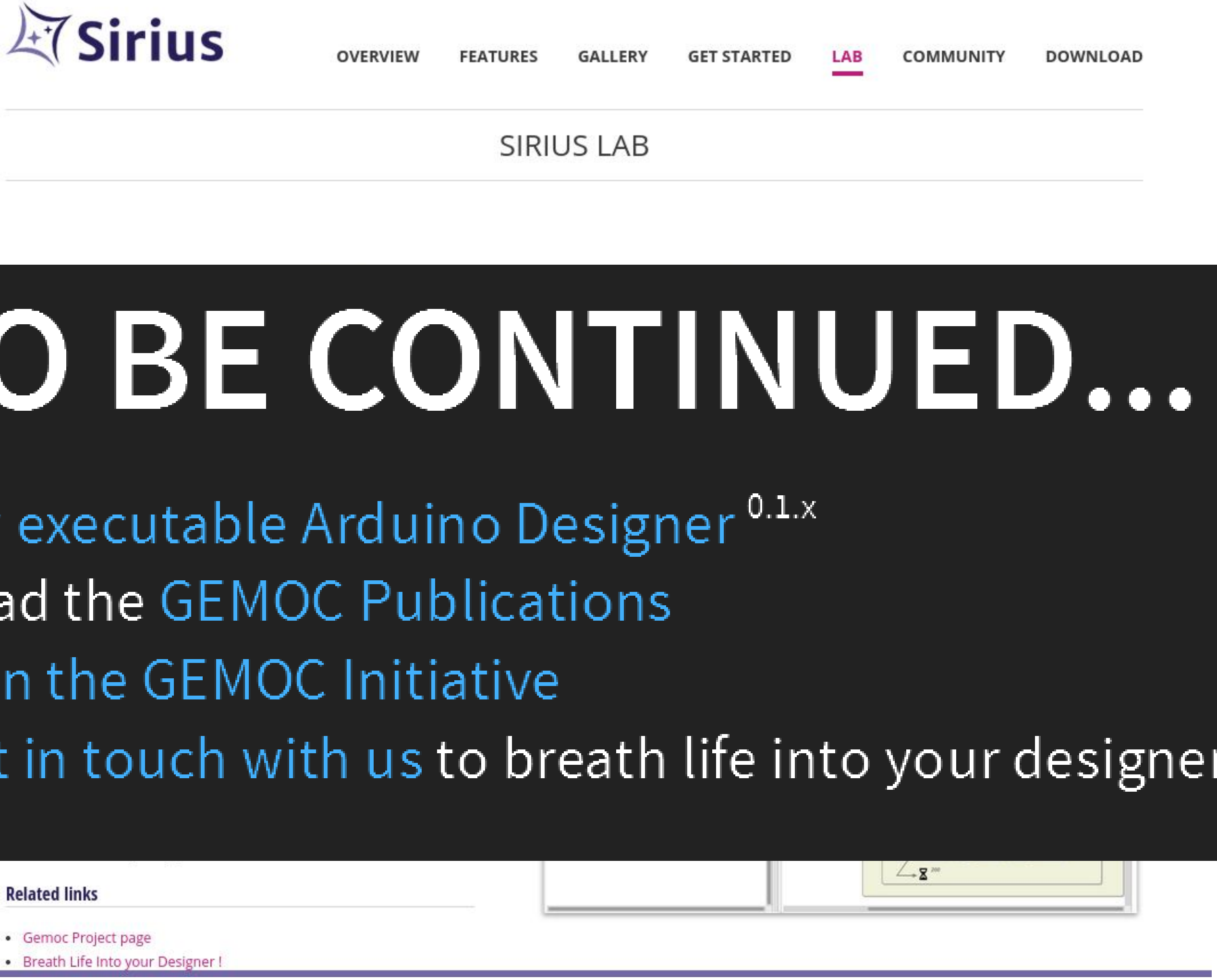
- **Marked graph tutorial** (cf. http://gemoc.github.io/gemoc-studio/publish/tutorial_markedgraph/html_single/GuideTutorialMarkedGraph.html)
- Activity Diagram (fUML) (cf. https://github.com/gemoc/activitydiagram)
- Arduino Modeling (cf. https://github.com/gemoc/arduinomodeling)
- Farming modeling (cf. https://github.com/gemoc/farmingmodeling)

Internal experimentation :

- xCapella, an executable extension of Capella (PoC)

Transfer

Generic technologies (EPL) which have proven helpful and are looking for an interest to bring it at full maturity level.



SIRIUS LAB

Related links

- Gemoc Project page
- Breath Life Into your Designer !

TO BE CONTINUED...

- Try executable Arduino Designer $^{0.1.x}$
- Read the GEMOC Publications
- Join the GEMOC Initiative
- Get in touch with us to breath life into your designer