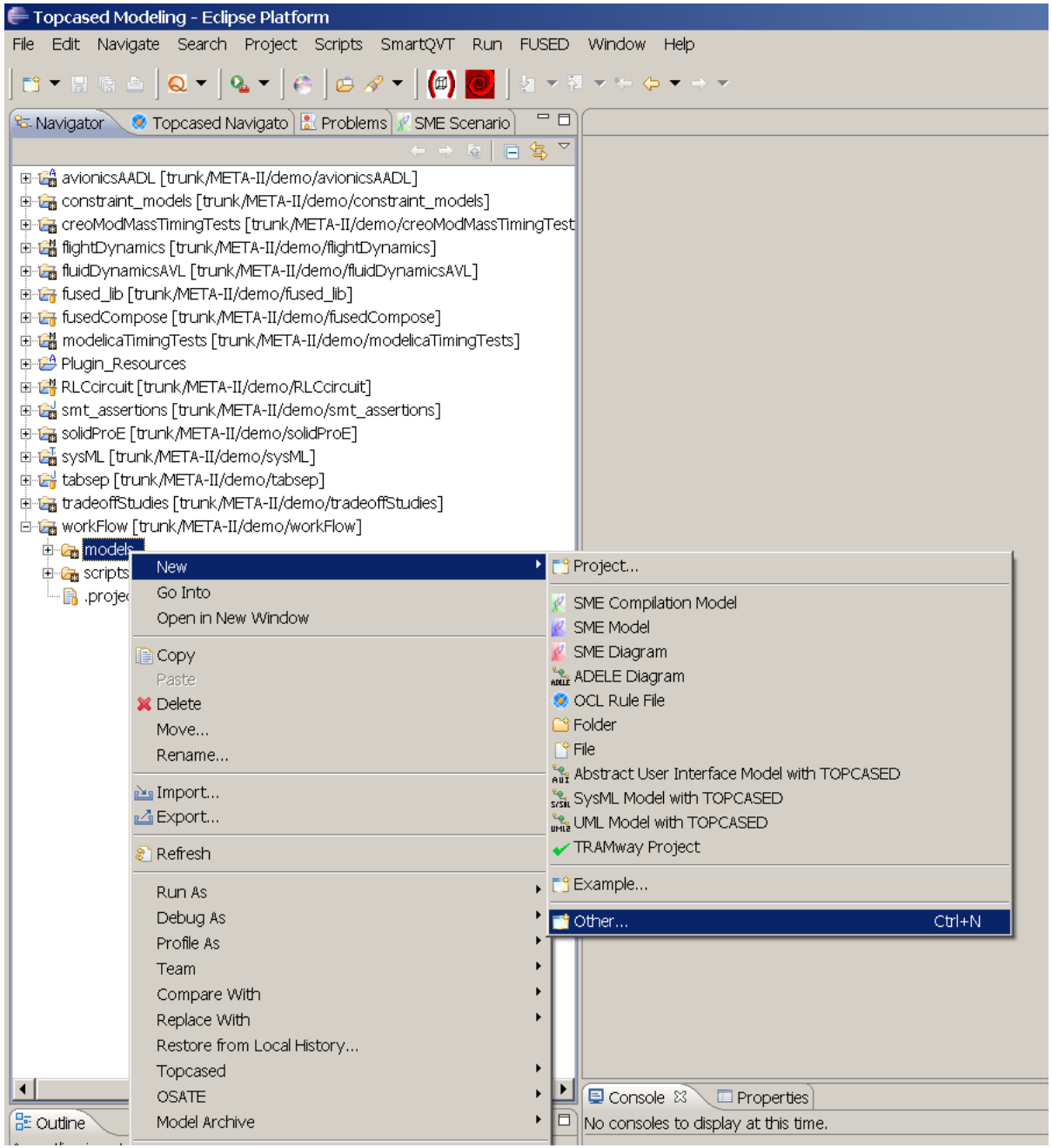
FUSED Demo
Graph Isomorphism

Generic, automated approach for comparing two or more models of diverse types.

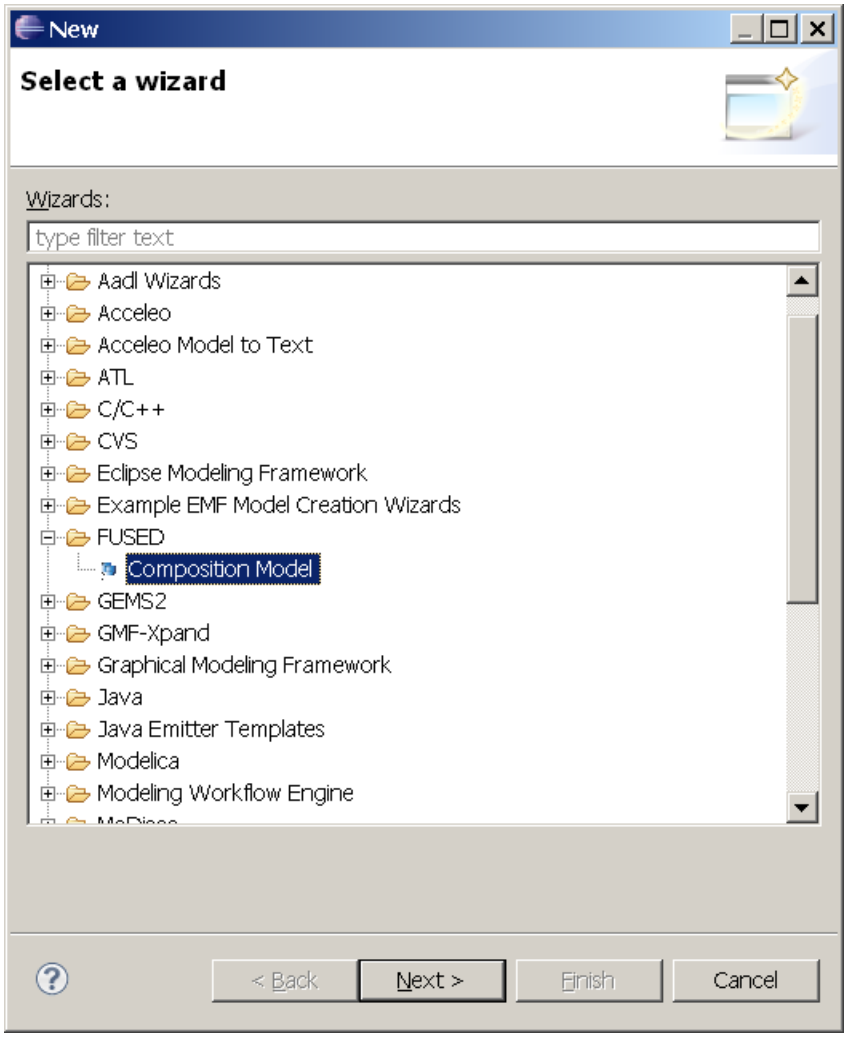
1) Create FUSED Composition Model

- In Navigator select *workFlow* project , *models* directory
- Select *New/Other*



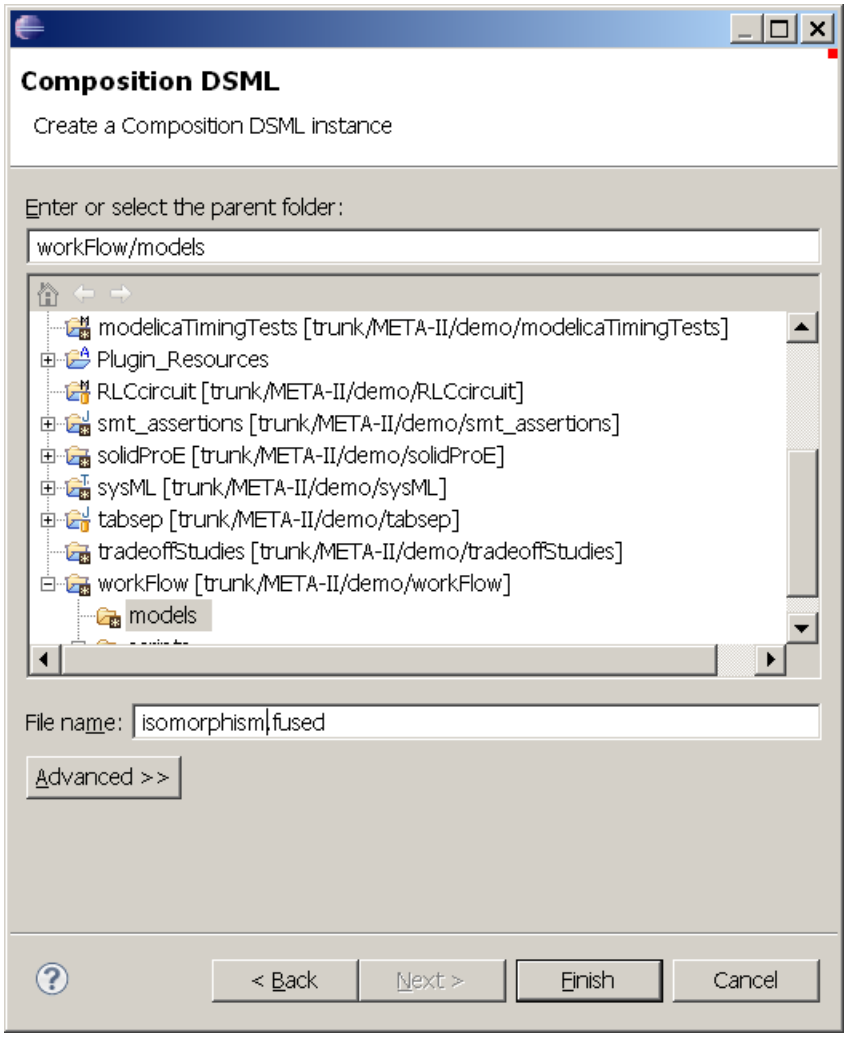
1) Create FUSED Composition Model

- In Navigator select *workFlow* project , *models* directory
- Select *New/Other*
- From the New Wizard window, select *FUSED/Composition Model*
- Select *Next*



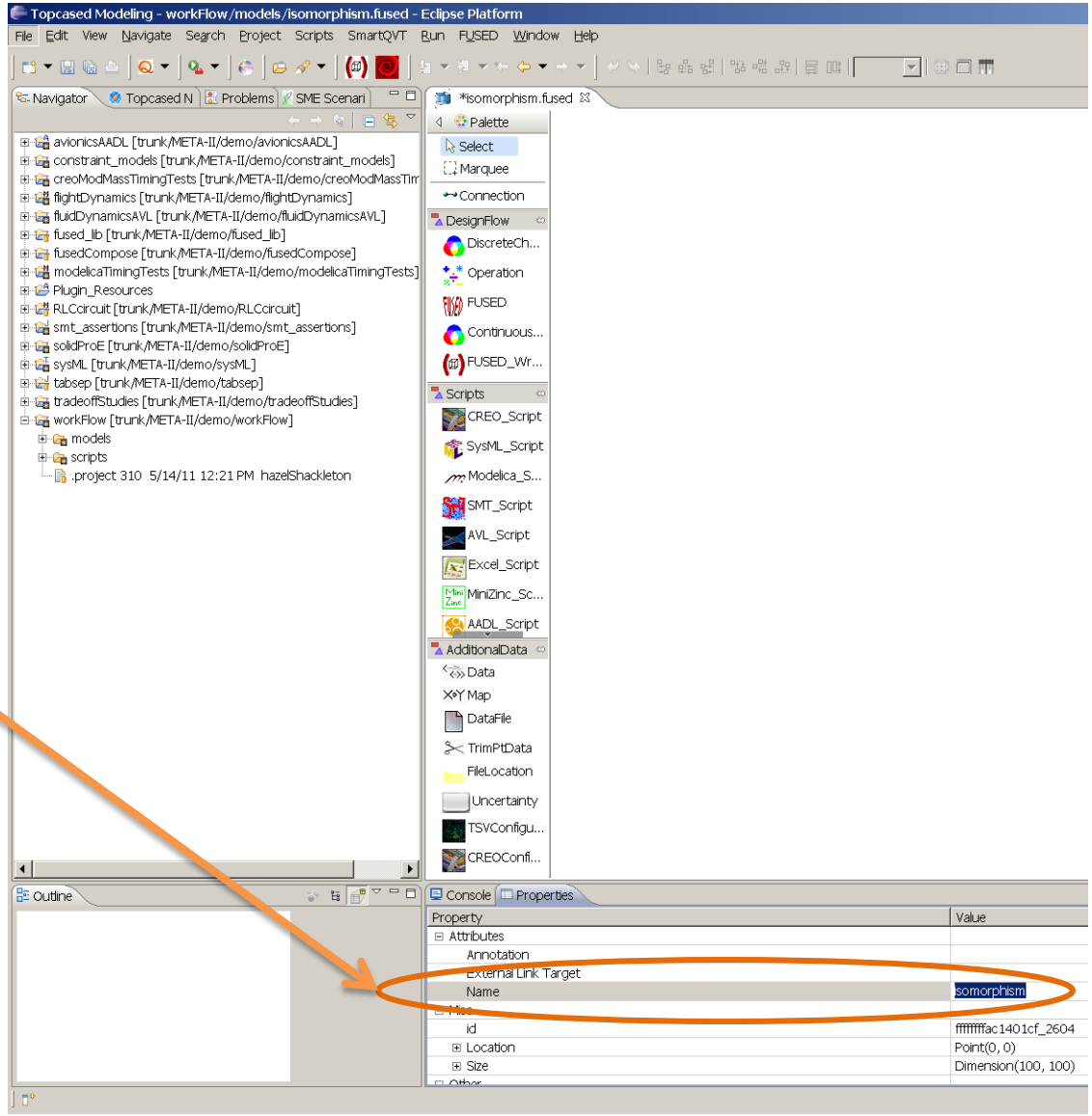
1) Create FUSED Composition Model

- In Navigator select *workFlow* project , *models* directory
- Select *New/Other*
- From the New Wizard window, select *FUSED/Composition Model*
- Select *Next*
- In Composition DSML window, change the File name (Note: all Fused Composition Models must end in ".fused")
- Select *Finish*



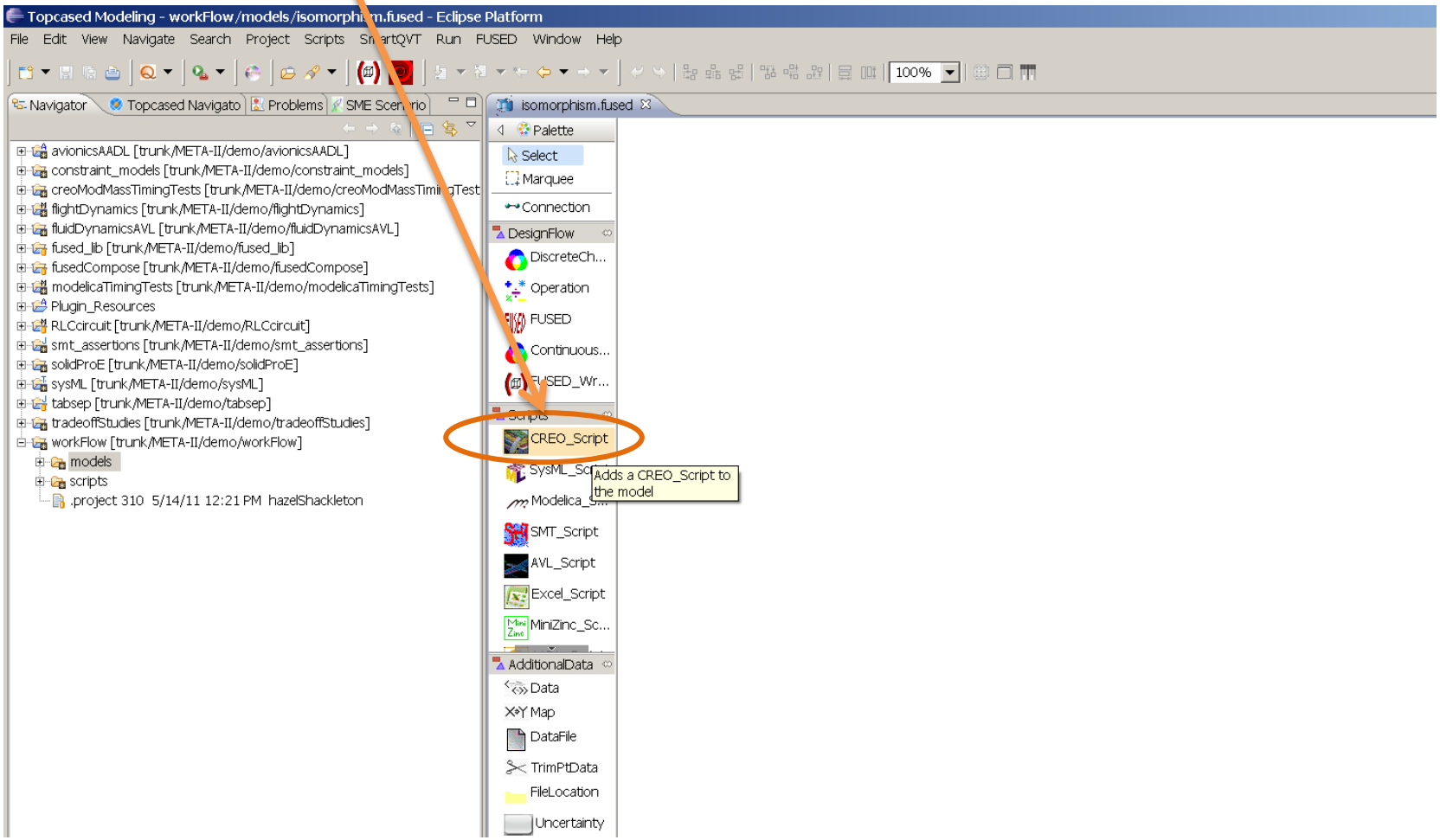
1) Create FUSED Composition Model

- In Navigator select *workFlow* project , *models* directory
- Select *New/Other*
- From the New Wizard window, select *FUSED/Composition Model*
- Select *Next*
- In Composition DSML window, change the File name (Note: all Fused Composition Models must end in ".fused")
- Select *Finish*
- Name the new Model using the Properties View (Note: the name of the model must match the file name without the ".fused" ending)—click in the main graphical view if the base properties are not visible.
- And Save the new Model (File/Save)



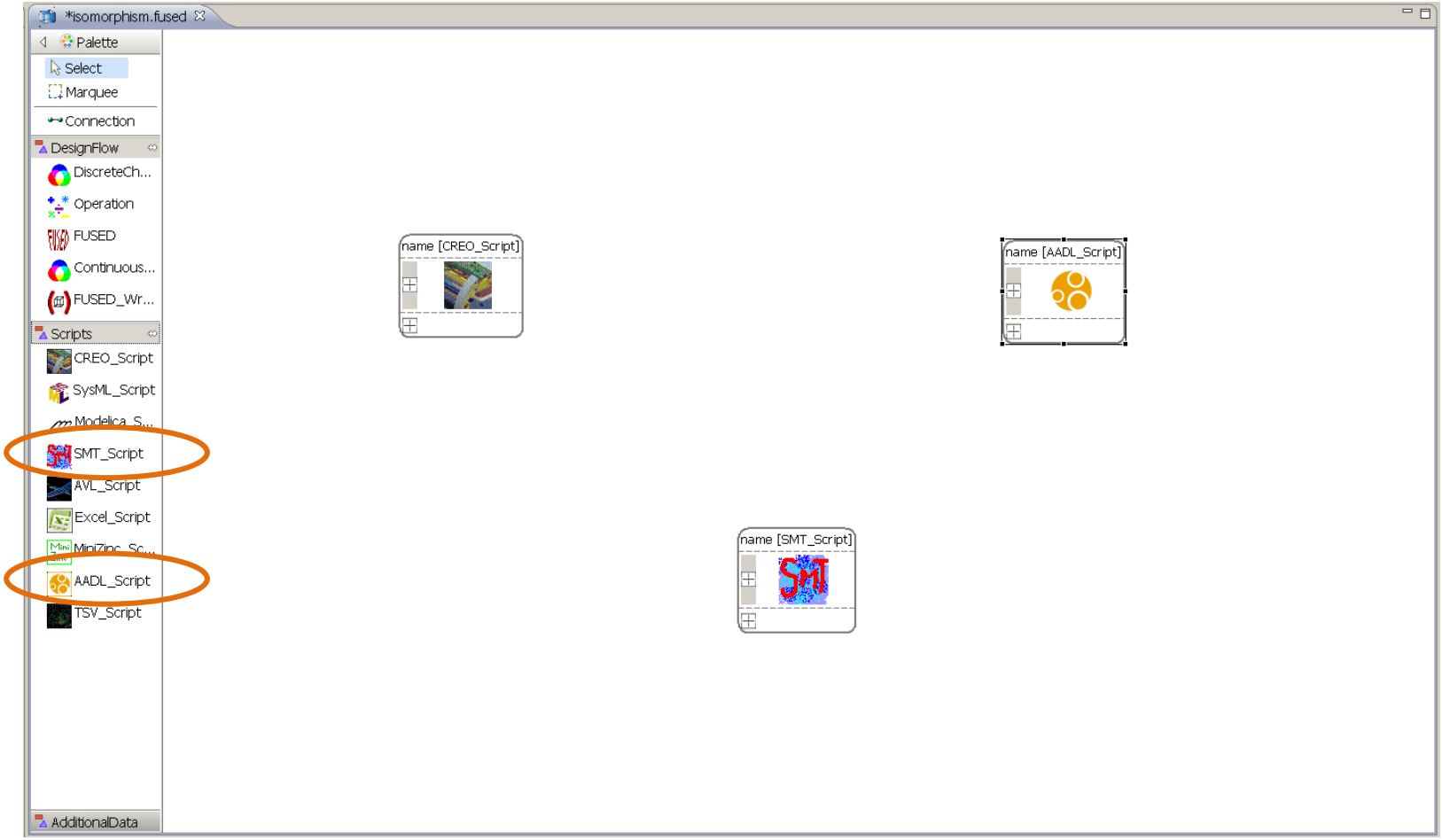
2) Define the Flow Configuration

- From the Palette View, select the Script desired (in this case *CREO_Script*) and click in the main view where you would like this component instantiated



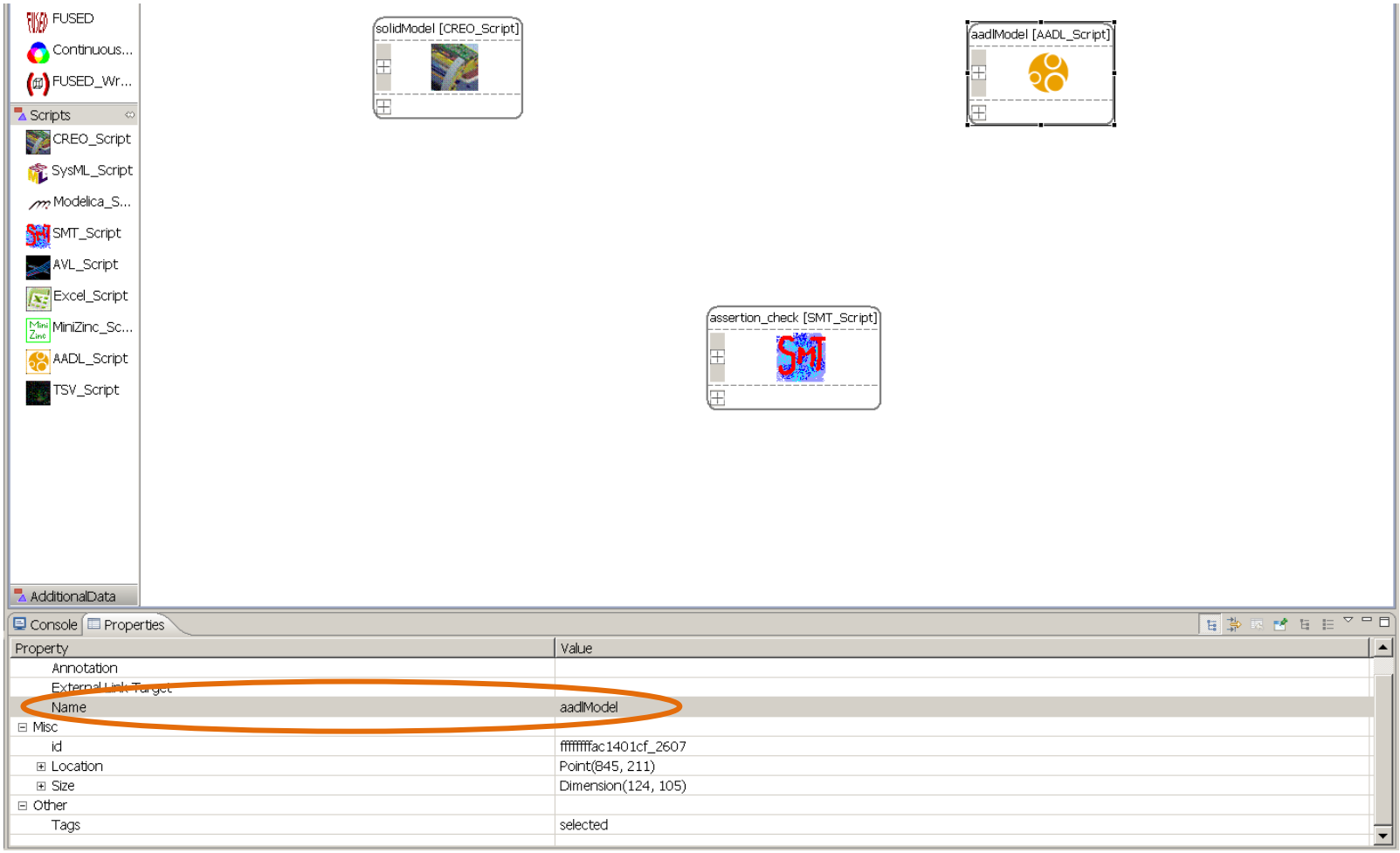
2) Define the Flow Configuration

- From the Palette View, select the Script desired (in this case *CREO_Script*) and click in the main view where you would like this component instantiated
- Repeat for the *AADL_Script* and *SMT_Script*



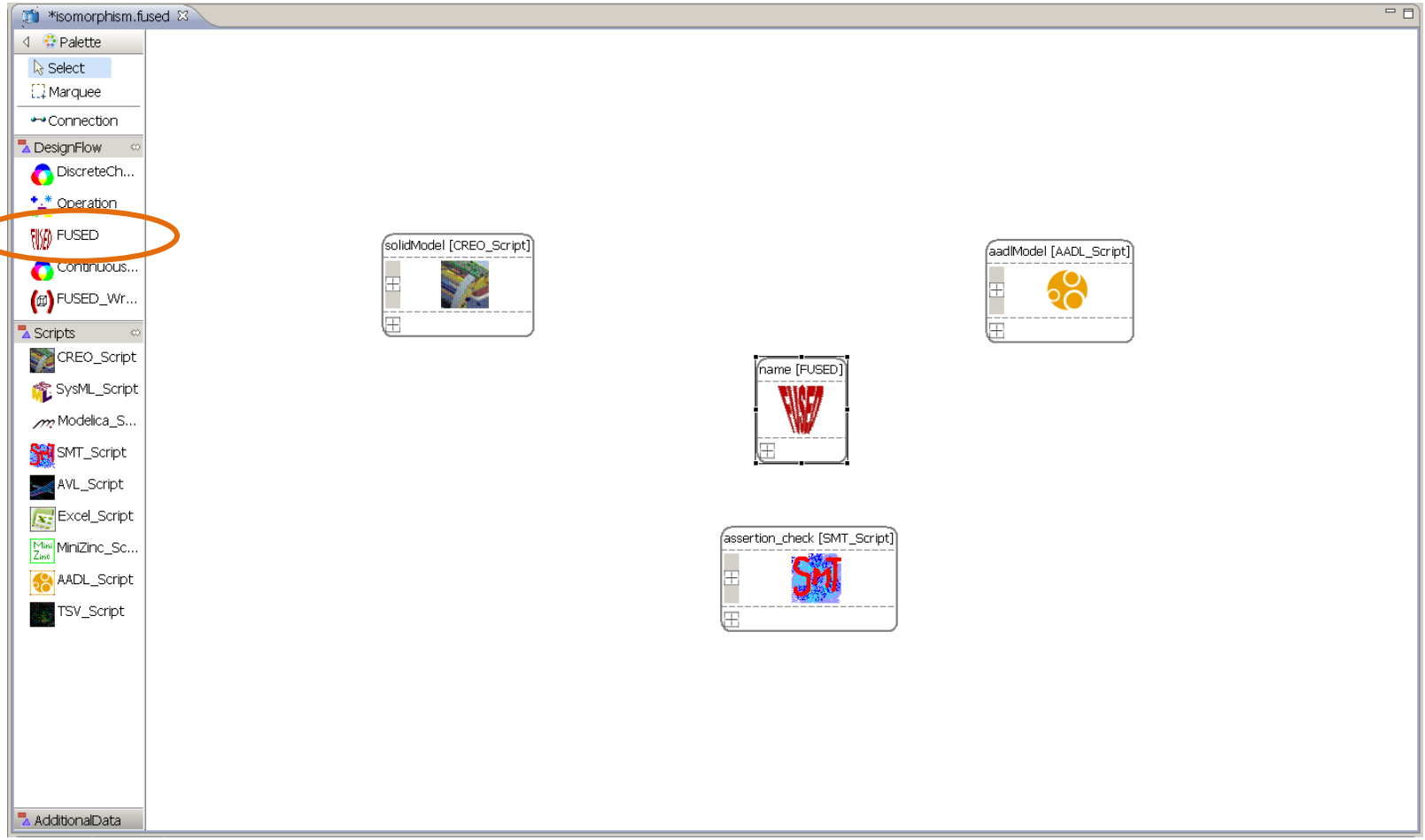
2) Instantiate Flow Components

- From the Palette View, select the Script desired (in this case *CREO_Script*) and click in the main view where you would like this component instantiated
- Repeat for the *AADL_Script* and *SMT_Script*
- Select and name each new component using the Property View (Note: these names are for readability only)



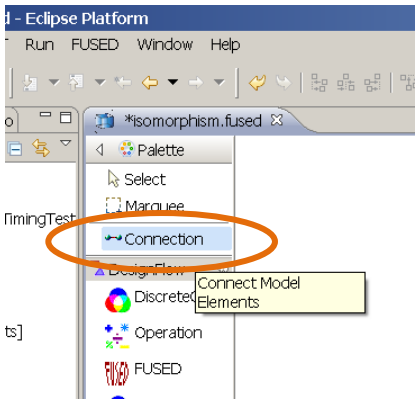
2) Instantiate Flow Components

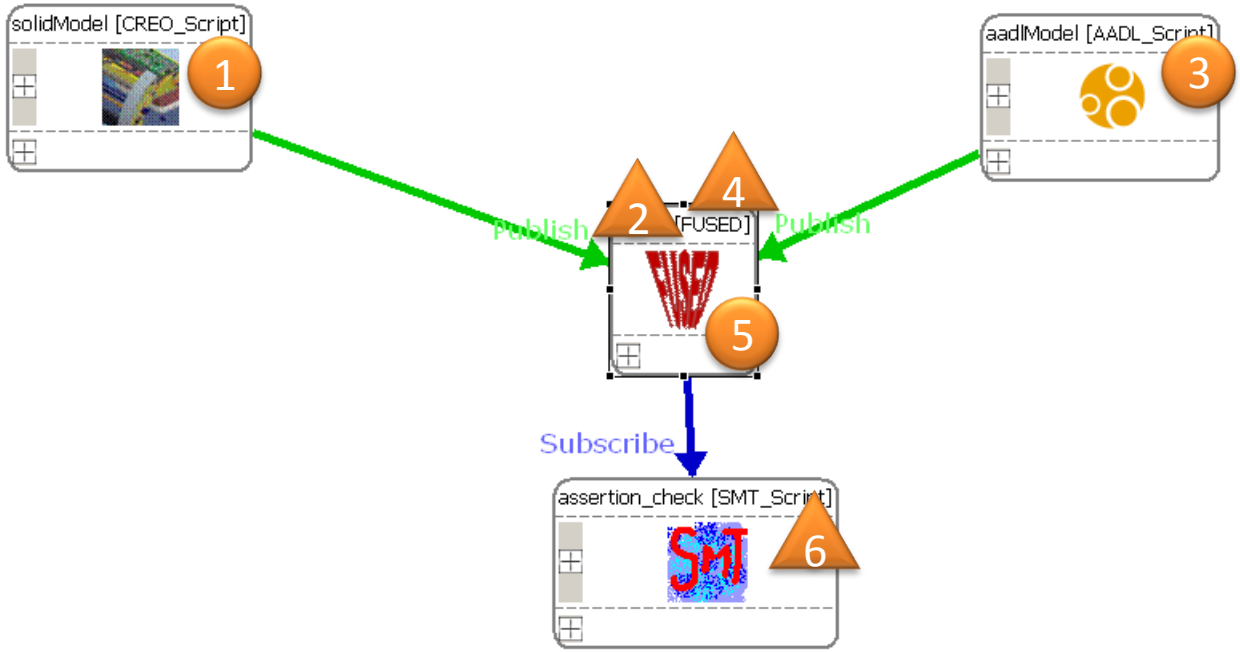
- From the Palette View, select the Script desired (in this case *CREO_Script*) and click in the main view where you would like this component instantiated
- Repeat for the *AADL_Script* and *SMT_Script*
- Select and name each new component using the Property View (Note: these names are for readability only)
- Add FUSED component



3) Define Flow

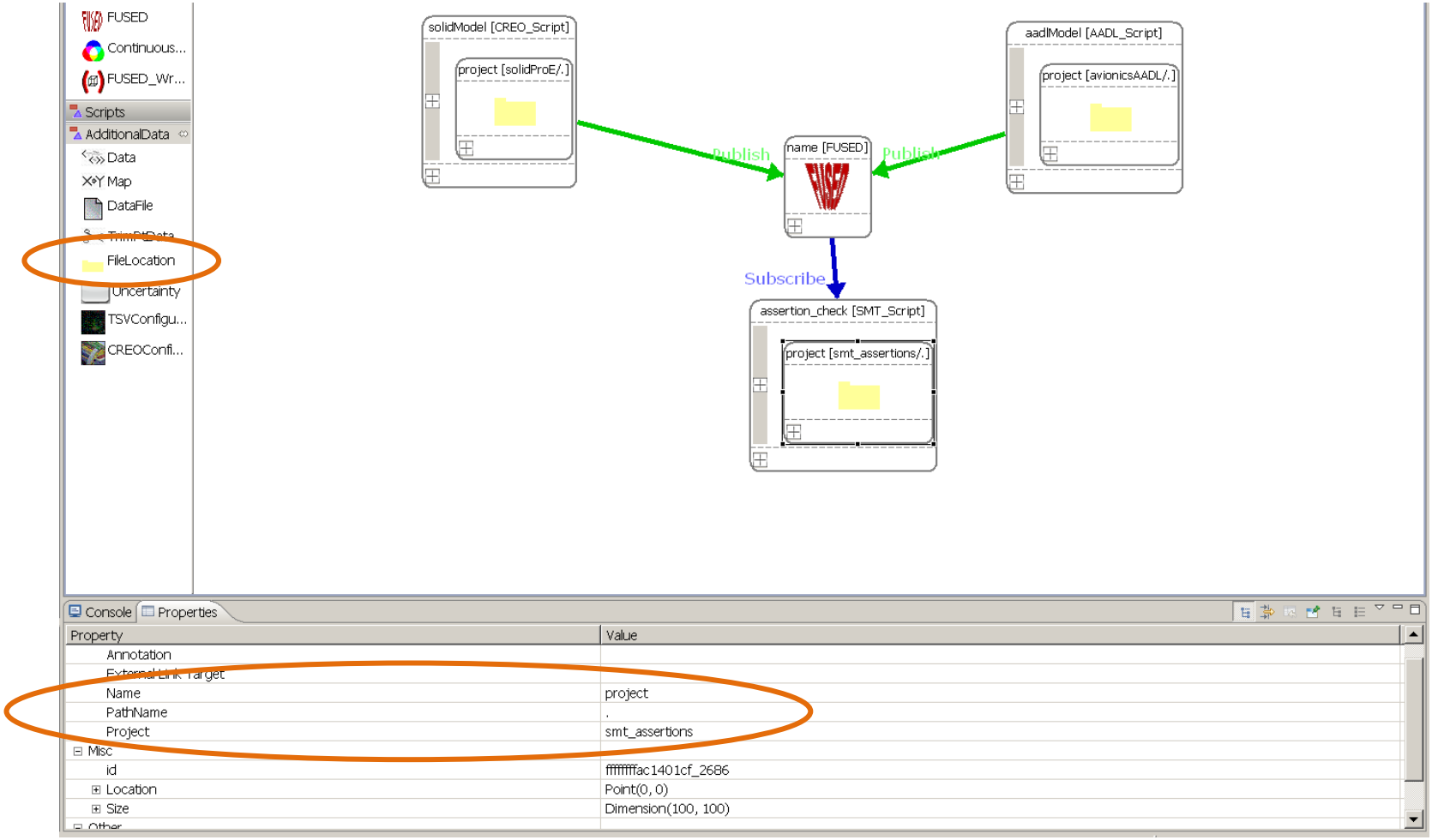
- Select the Connection element from the Palette
- Draw a publish connection between two components by selecting first the source, then the destination of each link—here we select the *CREO_Script* (1) then the *FUSED* component (2).
- Add the second publish connection by selecting the *AADL_Script* (3) then the *FUSED* component (4).
- Add a subscribe connection by starting at the *FUSED* component (5) and then selecting the *SMT_Script* (6).





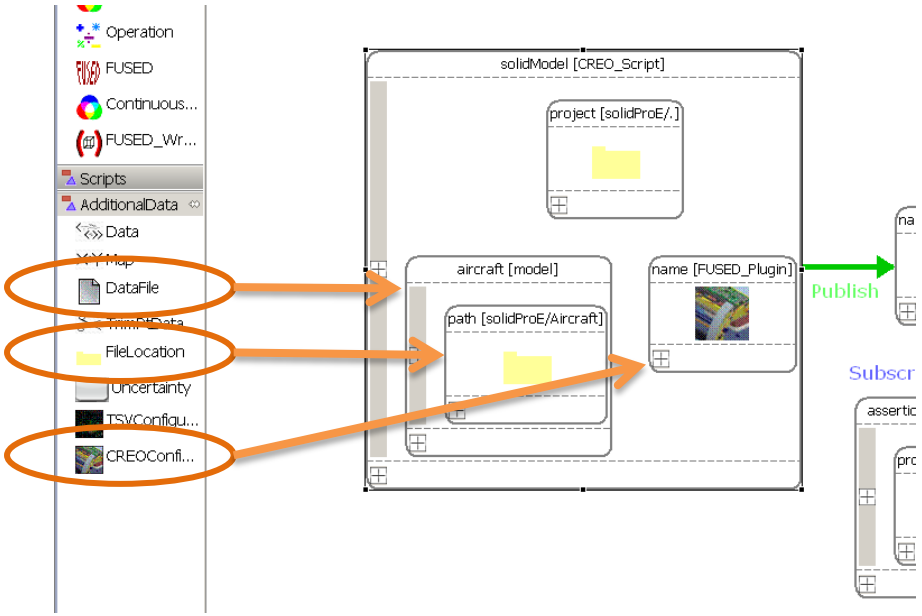
4) Add Locations to each Component

- From the Palette View, select the *FileLocation* icon and then select the script component in the main graphical view where you would like this component added
- Repeat for each of the other two script components
- Use the Property View to enter project and path names for each new *FileLocation* componenent created

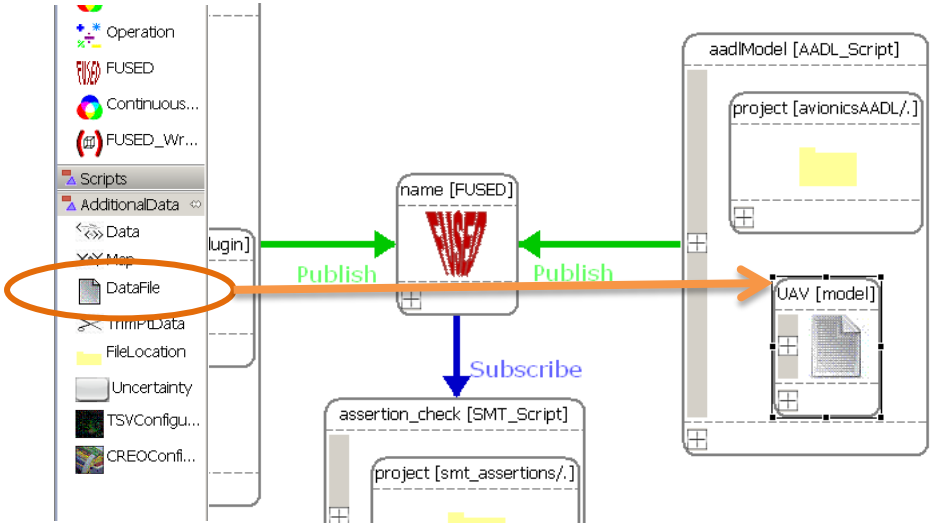


5) Complete Component Configuration

- For the *CREO_Script*, add a *DataFile* component and a *CREOConfiguration* component
- Use the Properties View to set the *DataFile* name and change the *FileType* to “Model” from the enumerations given
- Because the aircraft.asm model is located in a subdirectory within the project it will also need a *FileLocation*. Select *FileLocation* icon from the Palette, then click on the *DataFile/model* component. Set the project & path names in the Properties View

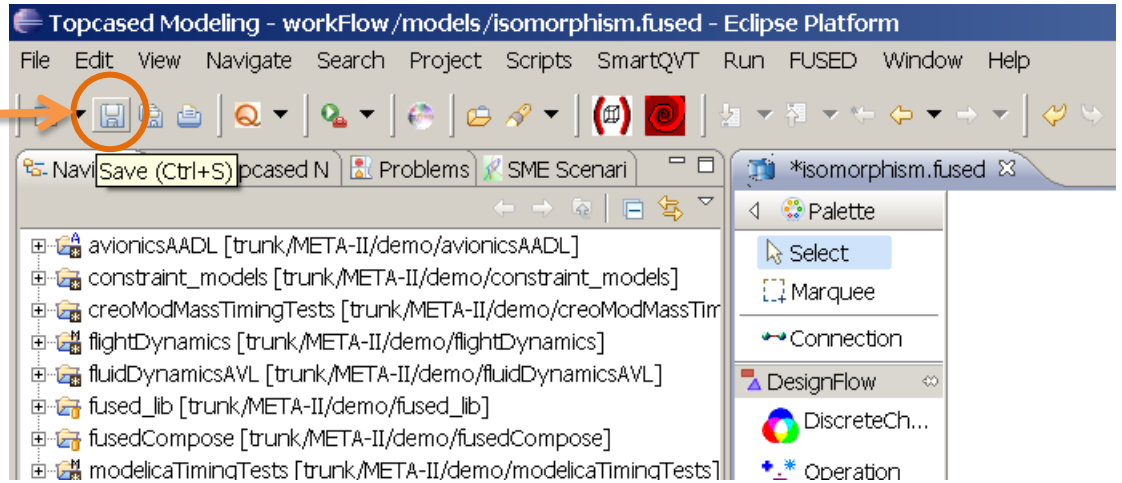


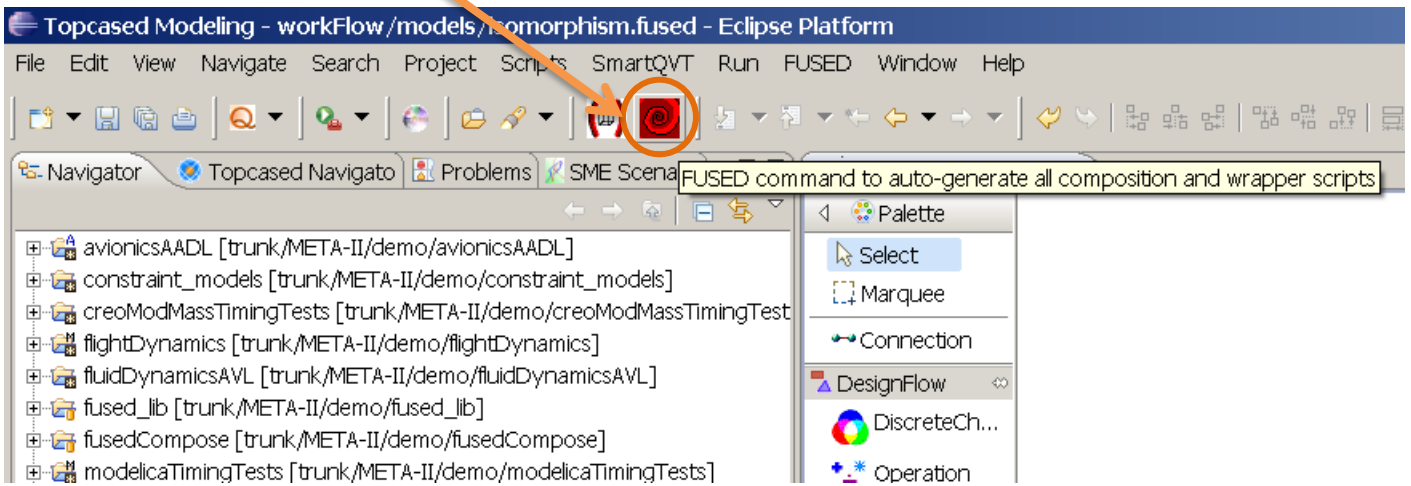
- For the AADL_Script, add a DataFile component
- Use the Properties View to set the DataFile name and change the FileType to “Model” from the enumerations given



6) Save and Run the Configuration

- Save the Model (you can also do this from the menu using *File/Save*)
- Run the Script (you can also do the from the menu using *FUSED/FUSED Generate Script Command*)

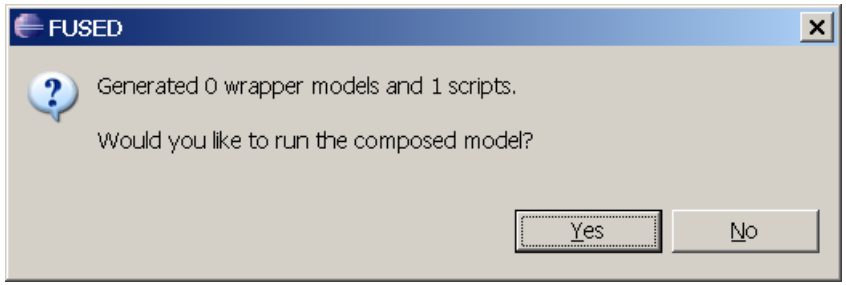


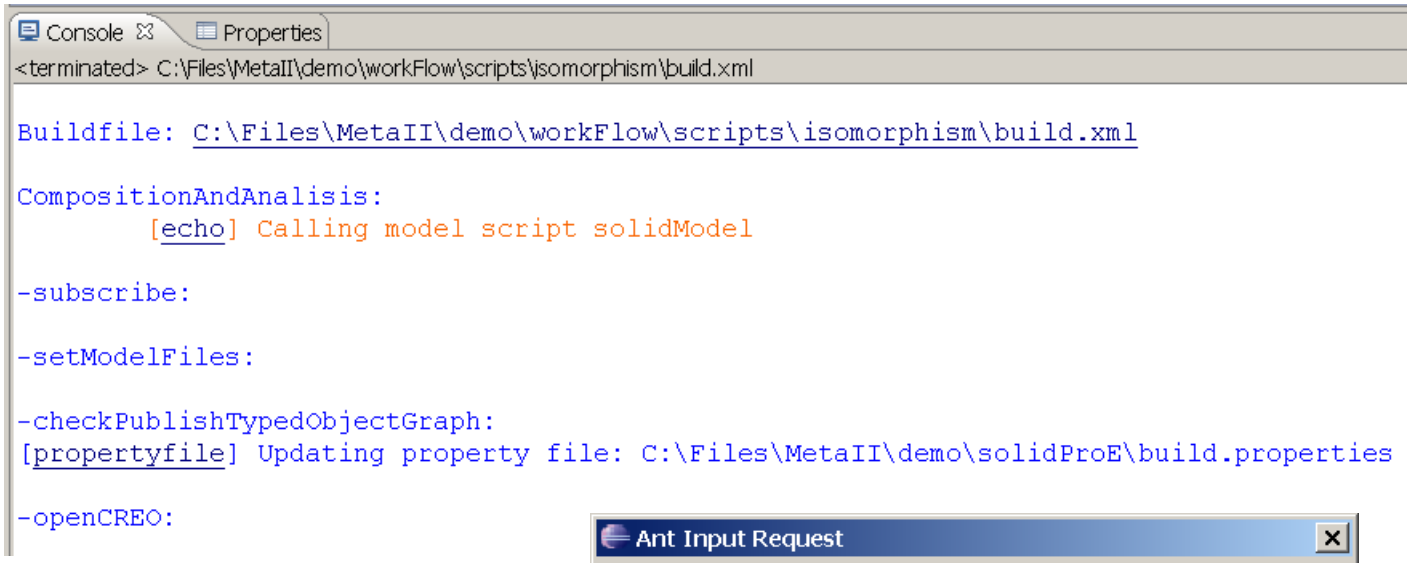


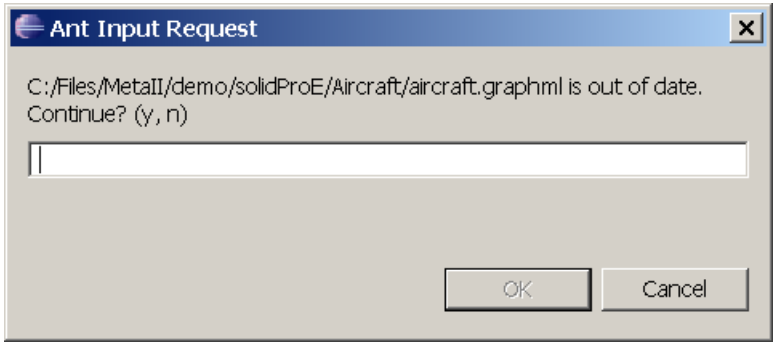
7) Composed Model Execution

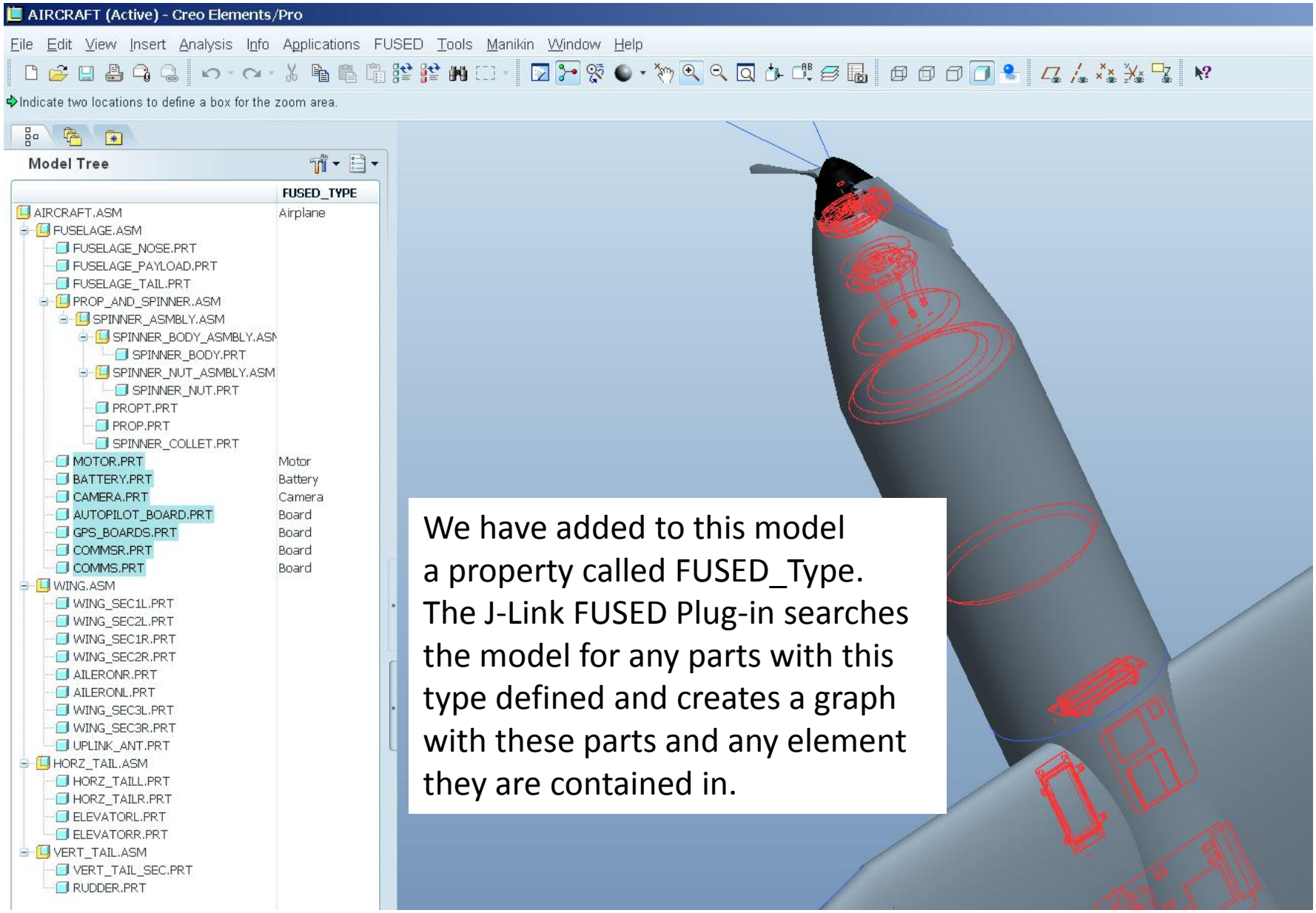
- One Script is generated, Select *Yes* to execute the composed model

- Watch the Console Window for actions taken—since this is the first execution, the ".graphML" file is out of date, wait for CREO to open



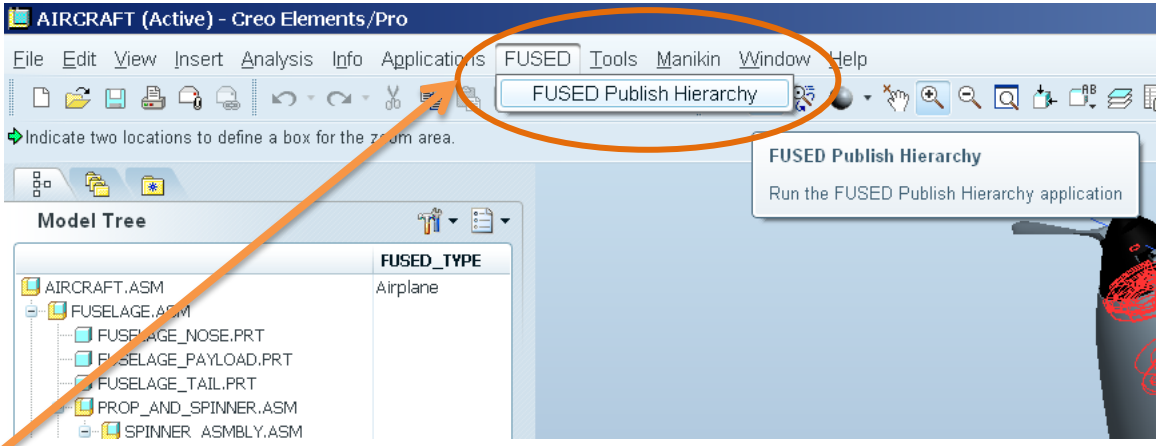




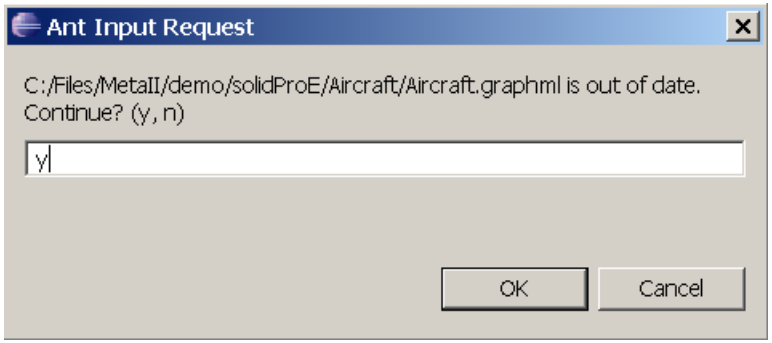

AIRCRAFT (Active) - Creo Elements/Pro
File Edit View Insert Analysis Info Applications FUSED Tools Manikin Window Help
Indicate two locations to define a box for the zoom area.
Model Tree
FUSED_TYPE
AIRCRAFT.ASM Airplane
FUSELAGE.ASM
FUSELAGE_NOSE.PRT
FUSELAGE_PAYLOAD.PRT
FUSELAGE_TAIL.PRT
PROP_AND_SPINNER.ASM
SPINNER_ASMBLY.ASM
SPINNER_BODY_ASMBLY.ASM
SPINNER_BODY.PRT
SPINNER_NUT_ASMBLY.ASM
SPINNER_NUT.PRT
PROPT.PRT
PROP.PRT
SPINNER_COLLET.PRT
MOTOR.PRT Motor
BATTERY.PRT Battery
CAMERA.PRT Camera
AUTOPILOT_BOARD.PRT Board
GPS_BOARDS.PRT Board
COMMSR.PRT Board
COMMS.PRT Board
WING.ASM
WING_SEC1L.PRT
WING_SEC2L.PRT
WING_SEC1R.PRT
WING_SEC2R.PRT
AILERONR.PRT
AILERONL.PRT
WING_SEC3L.PRT
WING_SEC3R.PRT
UPLINK_ANT.PRT
HORZ_TAIL.ASM
HORZ_TAILL.PRT
HORZ_TAILR.PRT
ELEVATORL.PRT
ELEVATORR.PRT
VERT_TAIL.ASM
VERT_TAIL_SEC.PRT
RUDDER.PRT
We have added to this model a property called FUSED_Type. The J-Link FUSED Plug-in searches the model for any parts with this type defined and creates a graph with these parts and any element they are contained in.

7) Composed Model Execution

- One Script is generated, Select *Yes* to execute the composed model
- Watch the Console Window for actions taken—since this is the first execution, the ".graphML" file is out of date, wait for CREO to open



- From CREO call the FUSED J-Link Plug-in from the Menu this will output a ".graphml" file
- Once the file has been created continue with the Composed Script—type "y" and select *OK*



7) Composed Model Execution

- <...>
- Check the output for consistency.

```
-openCREO:

publishTypedObjectGraph:
        [echo] Publishing typed object graph abstraction for aircraft C:/Files/MetaII/demo/solidProE/Aircraft
[propertyfile] Updating property file: C:\Files\MetaII\demo\workFlow\scripts\isomorphism\fused.properties
        [echo] Calling model script aadlModel

-subscribe:

-setModelFiles:

-checkPublishTypedObjectGraph:
[propertyfile] Updating property file: C:\Files\MetaII\demo\avionicsAADL\build.properties

publishTypedObjectGraph:
[propertyfile] Updating property file: C:\Files\MetaII\demo\workFlow\scripts\isomorphism\fused.properties
        [echo] Calling model script assertion_check

-subscribe:

-setModelFiles:

-checkVerifySubgraphIsomorphismh:

verifySubgraphIsomorphism:
        [exec] Result: 1
        [echo] The two models are consistent.
BUILD SUCCESSFUL
Total time: 32 seconds
```

7) SMT FUSED Model Review

- Subscription to 2 models
- Declaration of types we care about
- Mapping of AADL types to CREO FUSED_Type properties

isomorphism.fused | solidarch.smt2.fused

```
;; This is an SMTLib specification to verify that a subgraph of
;; the avionics architecture specification (written in AADL) is
;; subgraph isomorphic to the solid model (written in Creo).  For
;; example, every logical processor should have a mapping to
;; a physical card in the solid model.

(FUSED-subscribe AIRCRAFT)

(FUSED-subscribe UAV)

;; ASSERT INTER-MODEL CONSISTENCY CONDITION

;; The following specifies that there be a subgraph isomorphism of the
;; processor and bus subgraph in the logical avionics model into a
;; solid model subgraph of cards and cables.  This would be written by
;; a system engineer, or more probably someone skilled in FUSED and
;; SMT-LIB who would help write such complex consistency relations.
;; Note that, once specified, this consistency specification should be
;; fairly robust to changes in the models.  It would not need to be
;; tweaked as engineers tweak the specs, in fact it would hopefully be
;; fairly reusable across projects.

;; Specify the subgraph in the avionics model to be mapped.  This
;; is done here by specifying the types of objects that must have a
;; mapping to solid model objects.
(define-fun avionicsSubgraphObject ((ae UAV.Object)) Bool
   (or
       (UAV.ofType ae UAV.HardwarePlatform.PfpCPU.Fast)
       (UAV.ofType ae UAV.Communications.Radio)
       (UAV.ofType ae UAV.MotorControl.Motor)
       (UAV.ofType ae UAV.SensorData.IMU)
       (UAV.ofType ae UAV.SensorData.GPS)
       (UAV.ofType ae UAV.PayloadControl.Payload)
;;       (UAV.ofType ae UAV.ActuatorControl.Servo)   ;; Uncomment to introduce an inconsistency
))

;; Specify allowed mappings from avionics objects to solid objects.
;; This is done here by specifying which types of avionics objects
;; are compatible with (may be mapped to) which types of solid objects.
(define-fun typeCompatible ((ae UAV.Object) (se AIRCRAFT.Object)) Bool
    (or
       (and (UAV.ofType ae UAV.HardwarePlatform.PfpCPU.Fast) (AIRCRAFT.ofType se AIRCRAFT.Board))
       (and (UAV.ofType ae UAV.Communications.Radio) (AIRCRAFT.ofType se AIRCRAFT.Board))
       (and (UAV.ofType ae UAV.MotorControl.Motor) (AIRCRAFT.ofType se AIRCRAFT.Motor))
       (and (UAV.ofType ae UAV.SensorData.IMU) (AIRCRAFT.ofType se AIRCRAFT.Board))
       (and (UAV.ofType ae UAV.SensorData.GPS) (AIRCRAFT.ofType se AIRCRAFT.Board))
       (and (UAV.ofType ae UAV.PayloadControl.Payload) (AIRCRAFT.ofType se AIRCRAFT.Camera))
    ))
```

7) SMT FUSED Model Review

- Subscription to 2 models
- Declaration of types we care about
- Mapping of AADL types to CREO FUSED_Type properties

- Actual smarts to do the isomorphism (this part shouldn't change)

❖ This passed through the Silver SMT program and produced over 2500 lines of code for these small example graphs.

```
        (and (UAV.ofType ae UAV.MotorControl.Motor) (AIRCRAFT.ofType se AIRCRAFT.Motor))
        (and (UAV.ofType ae UAV.SensorData.IMU) (AIRCRAFT.ofType se AIRCRAFT.Board))
        (and (UAV.ofType ae UAV.SensorData.GPS) (AIRCRAFT.ofType se AIRCRAFT.Board))
        (and (UAV.ofType ae UAV.PayloadControl.Payload) (AIRCRAFT.ofType se AIRCRAFT.Camera))
    ))

;; The following asserts that there exists at least one subgraph
;; isomorphism from the avionics subgraph selected by avionicsSubgraphObject
;; into the solid model graph.  This is probably a general-purpose assertion
;; that can be reused, the model-specific conditions are captured in
;; the above functions.

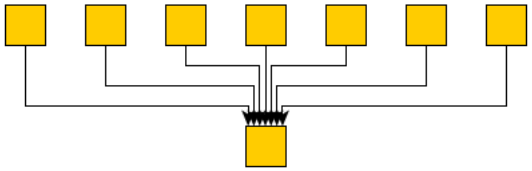
;; One condition that should be (easily) added is the ability for the
;; system engineer to explicitly specify a small number of seed
;; mappings, e.g. "avionics.mainProcessor maps to solid.mainCircuitCard".

;; The mapping from nodes in the above-specified avionics subgraph
;; to a solid model subgraph is defined by an array "map" whose index
;; is avionics objects and whose values are solid model objects.
;; The following condition essentially says there exists such a map
;; array that defines a subgraph isomorphism and satisfies the
;; above-defined object-to-object compatibility condition.
(assert
   (exists ((map (Array UAV.Object AIRCRAFT.Object)))
       (forall ((ae UAV.Object))
       (=>  (avionicsSubgraphObject ae)
        (and
           ; the map is 1-to-1, namely:
           ; if A maps to A' and B maps to A' then A = B
           (forall ((aeB UAV.Object))
               (=> (and (avionicsSubgraphObject aeB) (= (select map ae) (select map aeB)))
               (= ae aeB)))
           ; if A maps to A' and A attaches to B and B is in the subgraph
           ; then B maps to B' and A' attaches to B'
           (let ((se (select map ae)))
               (forall ((aeB UAV.Object))
               (=> (and (UAV.attached ae aeB UAV.connection) (avionicsSubgraphObject aeB))  ;; HARD
                   (AIRCRAFT.attached se (select map aeB) AIRCRAFT.Containment))))   ;; HARD-CODED
           ; if A maps to A'
           ; then class of A is compatible with class of A'
           (typeCompatible ae (select map ae)))))))

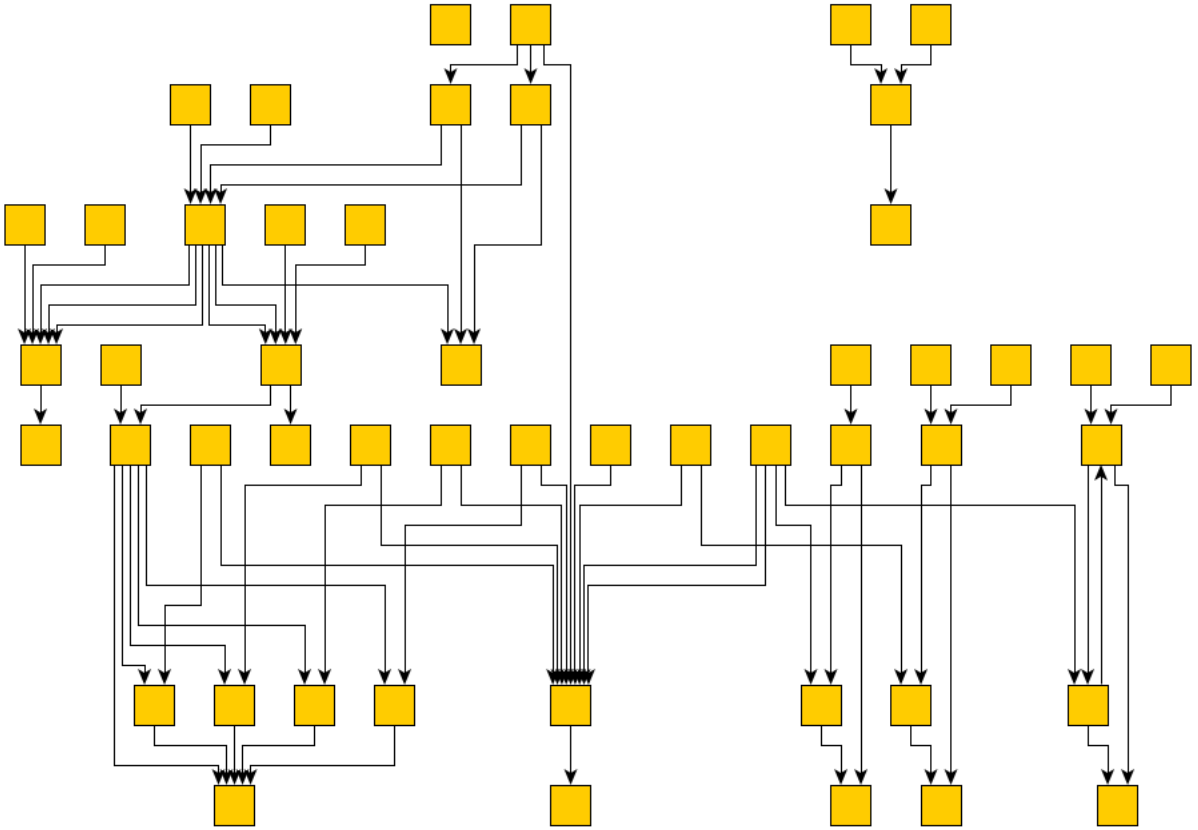
(check-sat)
(model)
```

8) Graph Reveiws

- Solid Model from CREO J-Link FUSED Plug-in



- Manually generated from AADL model



9a) Break Consistency (option #1)

- Open “solidarch.smt2.fused” from “smt_assertions” project
- Uncomment line 34 (delete “;;”) to add a new type of Servo that we require to have a match in both graphs.
- Save the file
- Close file
- Execute Model
- Check output

```
(define-fun avionicsSubgraphObject ((ae UAV.Object)) Bool
   (or
        (UAV.ofType ae UAV.HardwarePlatform.PfpCPU.Fast)
        (UAV.ofType ae UAV.Communications.Radio)
        (UAV.ofType ae UAV.MotorControl.Motor)
        (UAV.ofType ae UAV.SensorData.IMU)
        (UAV.ofType ae UAV.SensorData.GPS)
        (UAV.ofType ae UAV.PayloadControl.Payload)
;;       (UAV.ofType ae UAV.ActuatorControl.Servo)   ;; Uncomment to introduce an inco
))
```



```
-subscribe:

-setModelFiles:

-checkPublishTypedObjectGraph:
[propertyfile] Updating property file: C:\Files\MetaII\demo\solidProE\build.properties

publishTypedObjectGraph:
        [echo] Publishing typed object graph abstraction for aircraft C:/Files/MetaII/demo/solidProE/Aircraft
[propertyfile] Updating property file: C:\Files\MetaII\demo\workFlow\scripts\graphComparison\fused.properties
        [echo] Calling model script published_dynamics_model

-subscribe:

-setModelFiles:

-checkPublishTypedObjectGraph:
[propertyfile] Updating property file: C:\Files\MetaII\demo\avionicsAADL\build.properties

publishTypedObjectGraph:
[propertyfile] Updating property file: C:\Files\MetaII\demo\workFlow\scripts\graphComparison\fused.properties
        [echo] Calling model script graph_comparison

-subscribe:

-setModelFiles:

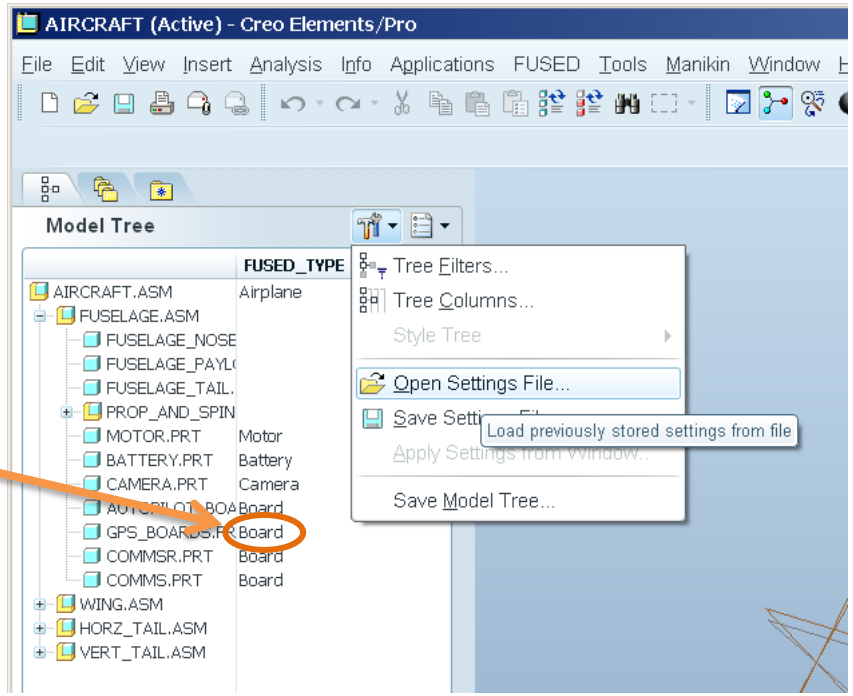
-checkVerifySubgraphIsomorphismh:

verifySubgraphIsomorphism:
      [echo] THE TWO MODELS ARE INCONSISTENT.
BUILD SUCCESSFUL
Total time: 17 seconds
```

9b) Break Consistency (option #2)

- Delete generated graph from solidProE project ("AIRCRAFT.graphml")
- Execute Model
- When CREO opens, open "tree.cfg" settings file
- Delete one of the FUSED_Type definitions (select, hit Delete,and *Yes* to confirm)
- Run the FUSED Plug-In from the Menu (FUSED/FUSED Publish Hierarchy)
- Continue the Composition Execution ("y" & OK)
- Check output






```
-checkPublishTypedObjectGraph:
[propertyfile] Updating property file: C:\Files\MetaII\demo\avionicsAADL\build.properties

publishTypedObjectGraph:
[propertyfile] Updating property file: C:\Files\MetaII\demo\workFlow\scripts\graphComparison\fused.properties
        [echo] Calling model script graph_comparison

-subscribe:

-setModelFiles:

-checkVerifySubgraphIsomorphismh:

verifySubgraphIsomorphism:
        [echo] THE TWO MODELS ARE INCONSISTENT.
BUILD SUCCESSFUL
Total time: 4 seconds
```