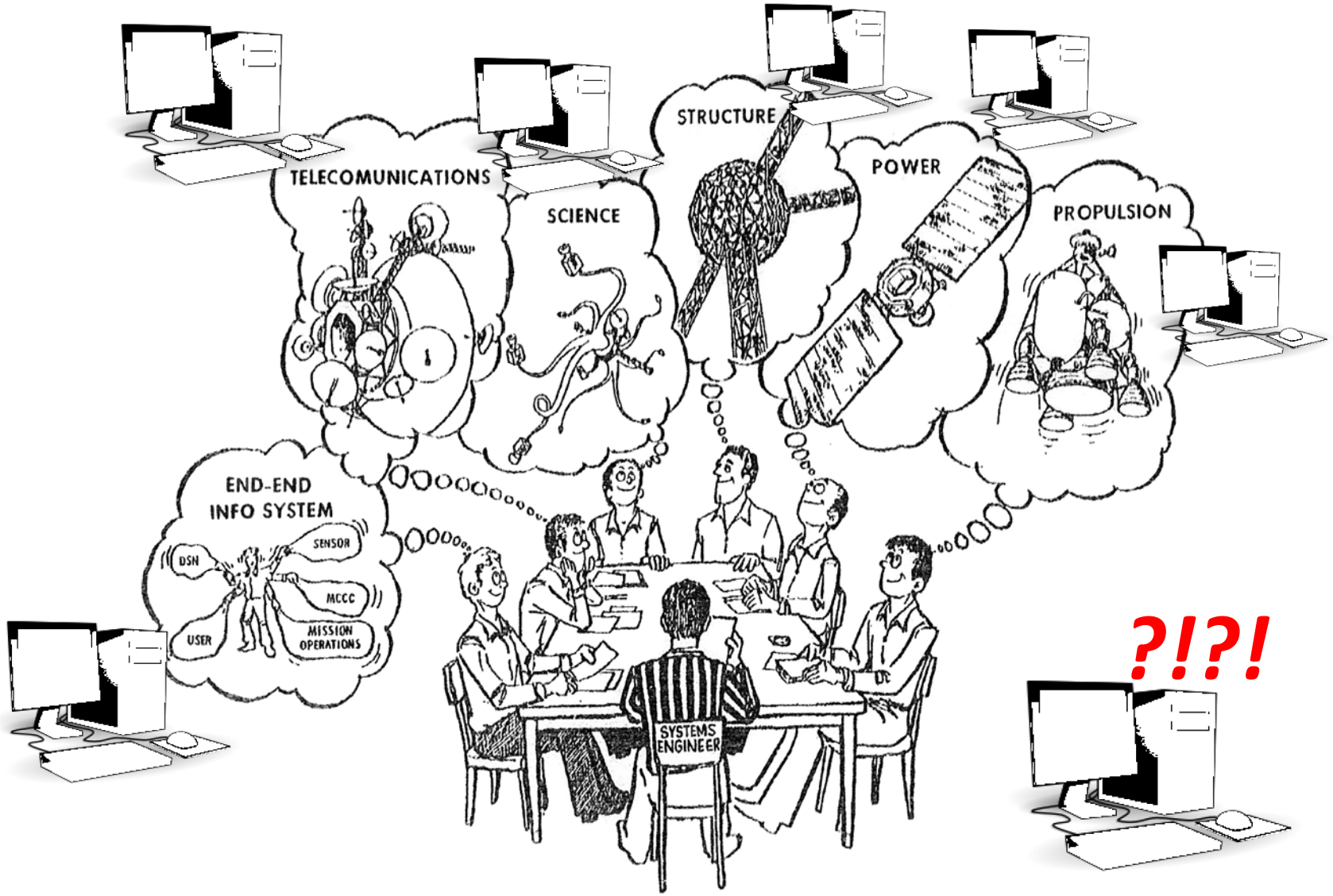
Getting a Cohesive Answer from a Common Start: Scalable Multidisciplinary Analysis through Transformation of a Systems Model

Bjorn Cole, Ph.D.
Seung Chung, Ph.D.
Jet Propulsion Laboratory,
California Institute of Technology

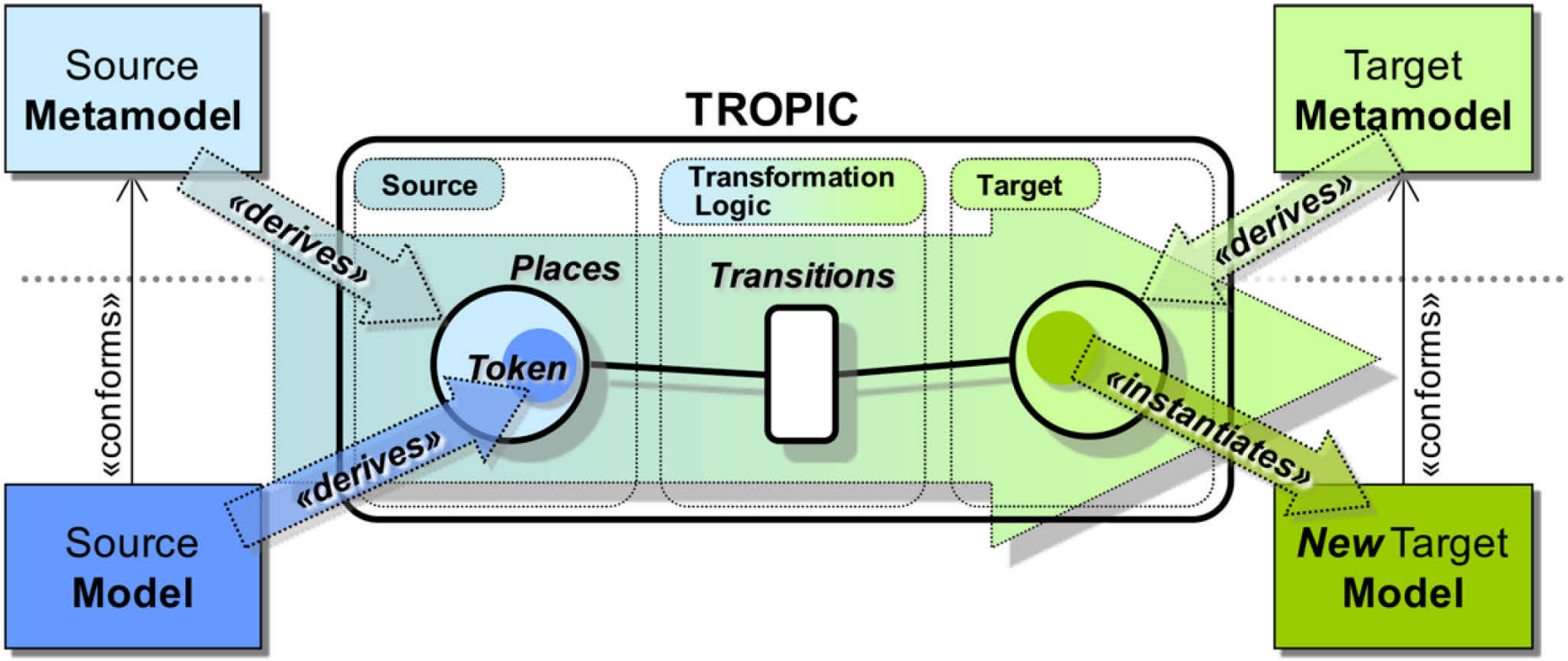
Bjorn.cole@jpl.nasa.gov

The Problem


TELECOMUNICATIONS
SCIENCE
STRUCTURE
POWER
PROPULSION
END-END
INFO SYSTEM
DSN
SENSOR
MCCC
USER
MISSION
OPERATIONS
SYSTEMS
ENGINEER
?!?!

Transformation in General

Image from webpage of Business Informatics Group, Vienna University of Technology



What QVTo looks like

A simple transformation example (from The Art of Model Transformation with Operational QVT - Sergey Boyko, Radomil Dvorak, Alexander Igdalov; March 2009)

From Publically Available Eclipse 2009 presentation

```
modeltype ECORE uses
    'http://www.eclipse.org/emf/2002/Ecore';
modeltype UML uses
    'http://www.eclipse.org/uml2/2.0.0/UML';

transformation Ecore2Uml(
    in inModel : ECORE, out outModel : UML);

main() {
    inModel.rootObjects()[EPackage]->map
    ePackage2Model();
}

mapping EPackage::ePackage2Model() : Model {
    name := self.name;
}
```

metamodels

metamodel URI in the EMF package registry

transformation signature

model parameters

entry point

mappings, etc.

JEO: SysML-Simulink Power Simulation

Represents hardware in its entirety, including its *parameters values*, and its specified *behavior*, such as *states*

Represents a broadcast *trigger* to cause the hardware to *change state* and thus consume a different amount of power

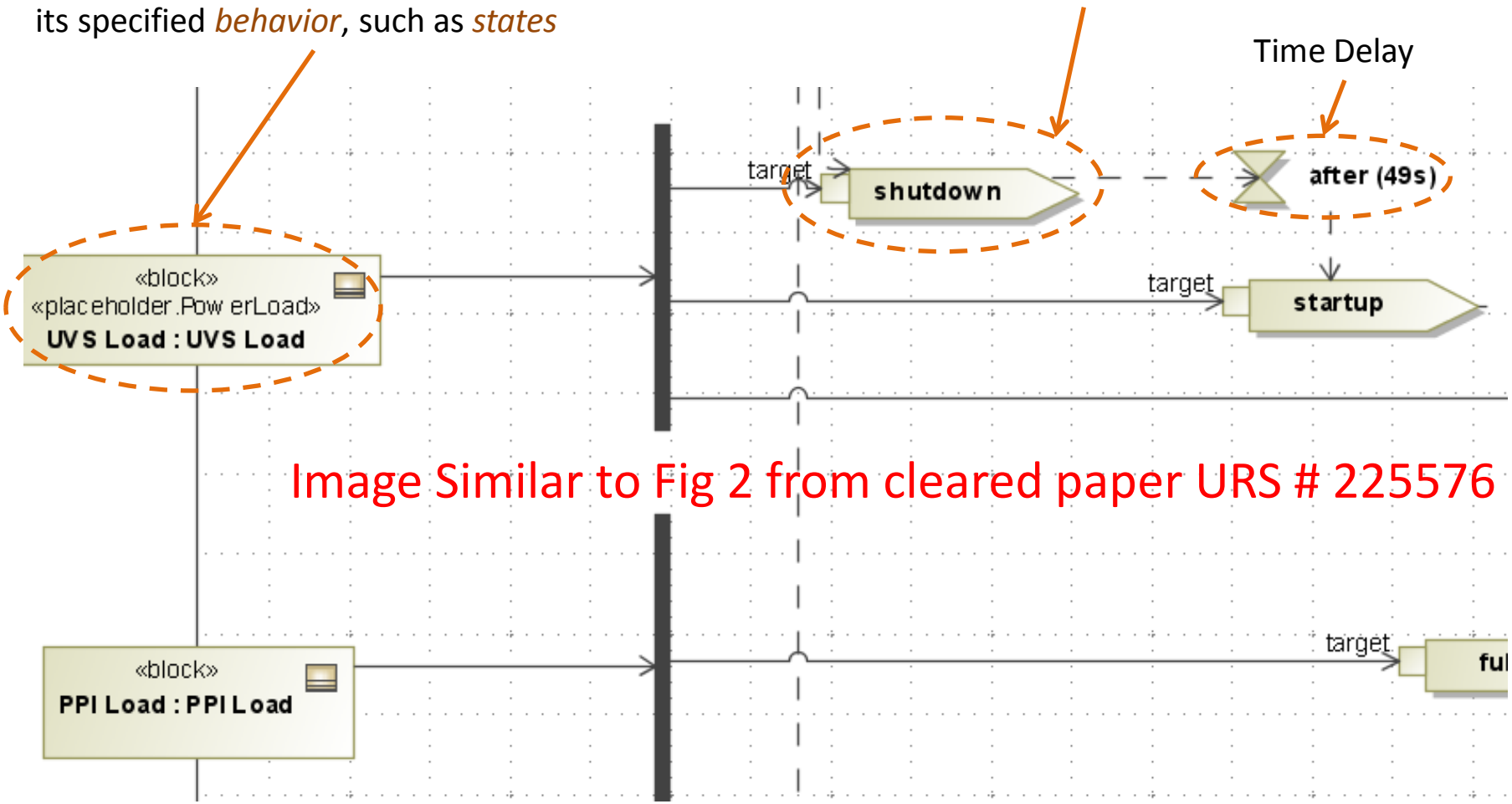
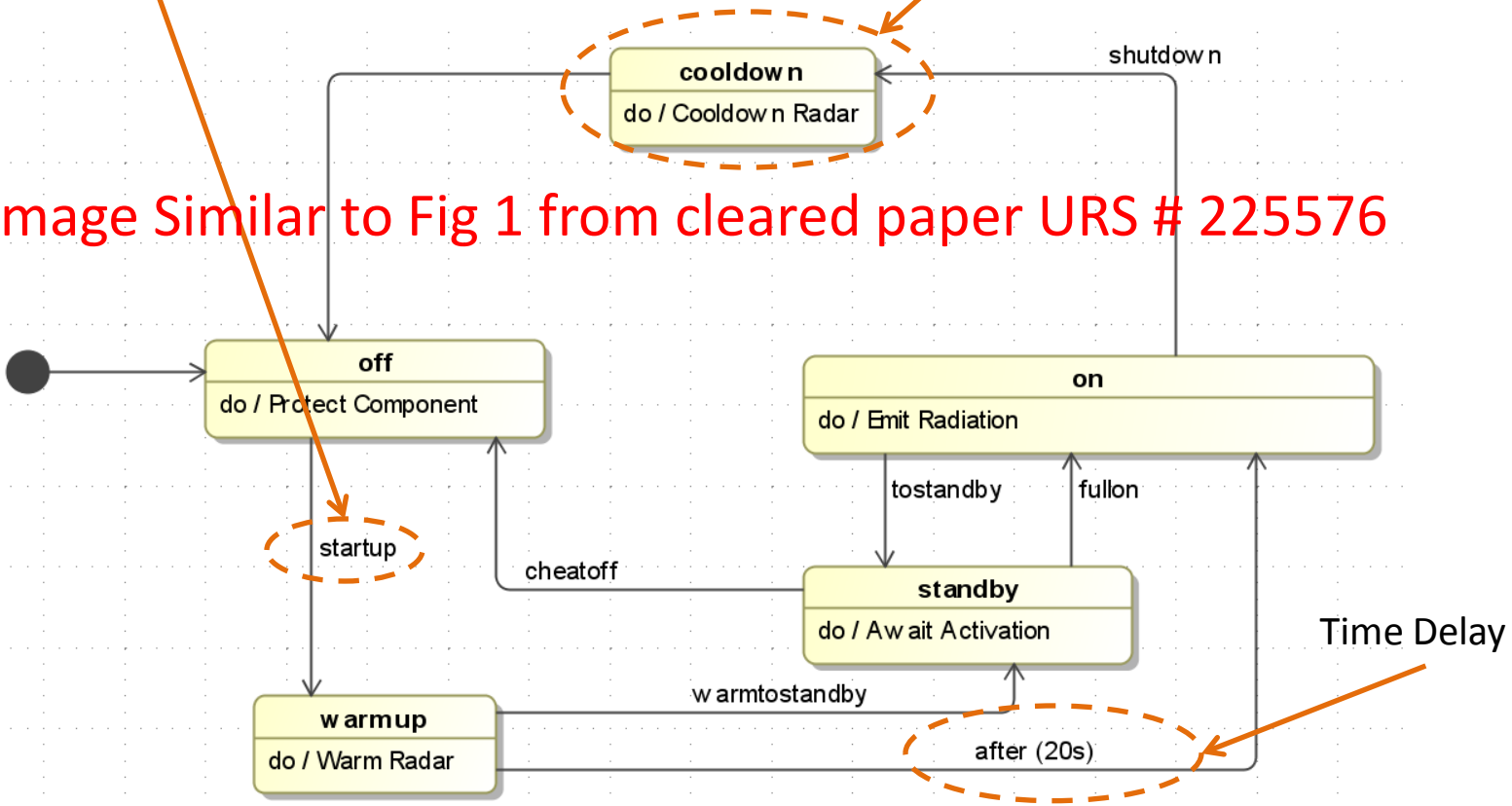


Image Similar to Fig 2 from cleared paper URS # 225576

JEO: SysML-Simulink Power Simulation

The triggers from the last page are mapped to *transitions* between *states*

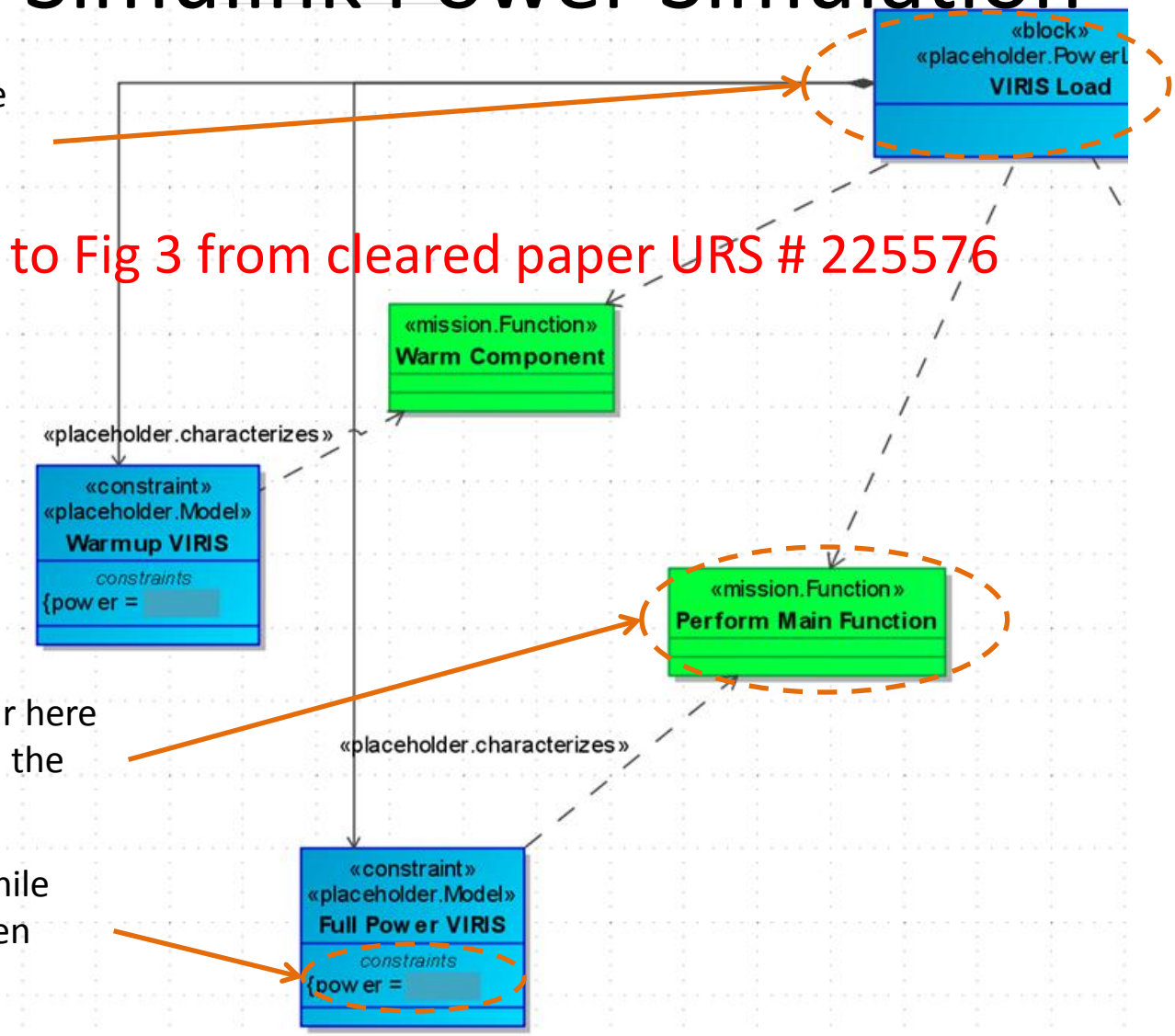
Each state contains a reference to what the *component should be doing* at this time, e.g., cooldown



JEO: SysML-Simulink Power Simulation

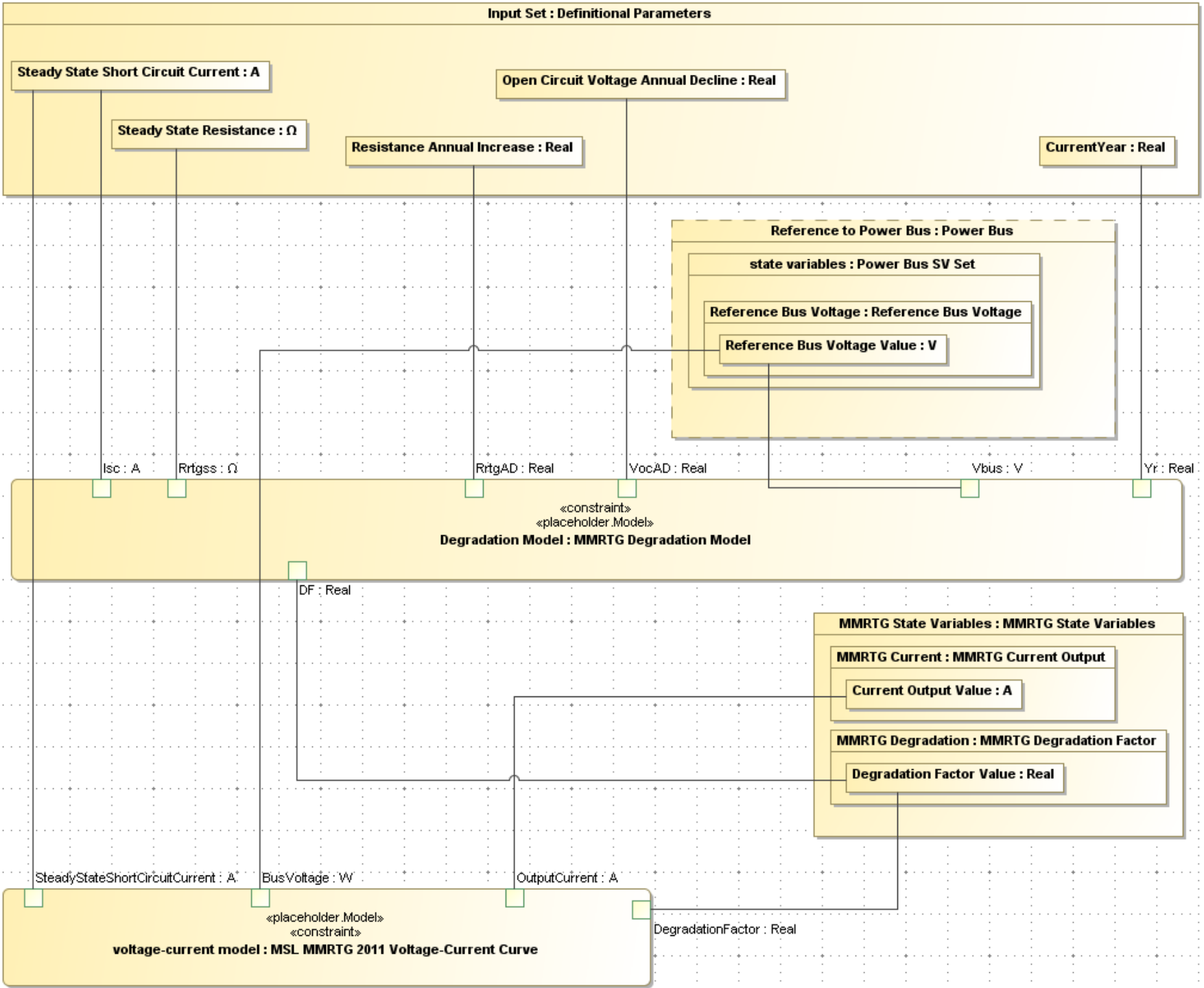
The component definition is the key to orchestrating all of these views

Image Similar to Fig 3 from cleared paper URS # 225576

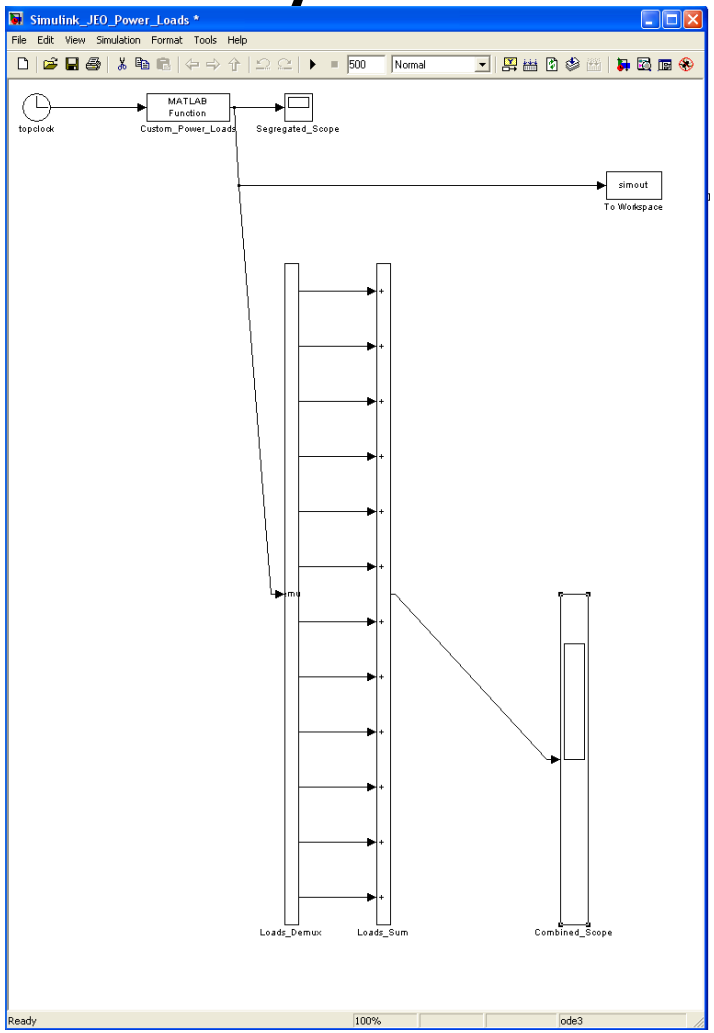


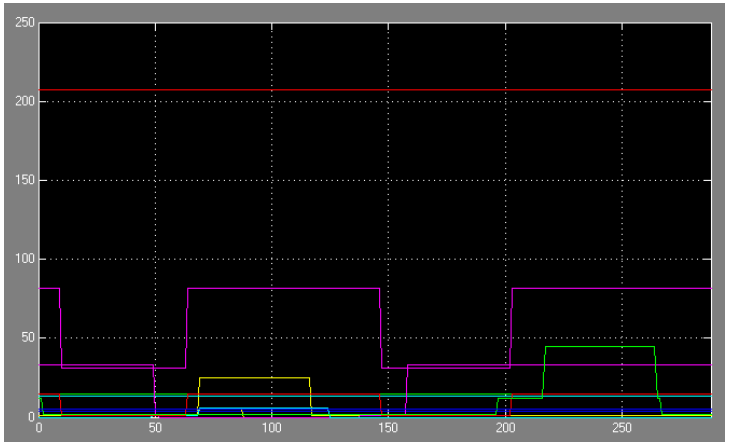
The functions performed appear here and are the same as those from the State Diagram

Finally, the power consumed while performing each function is given here, completing the map

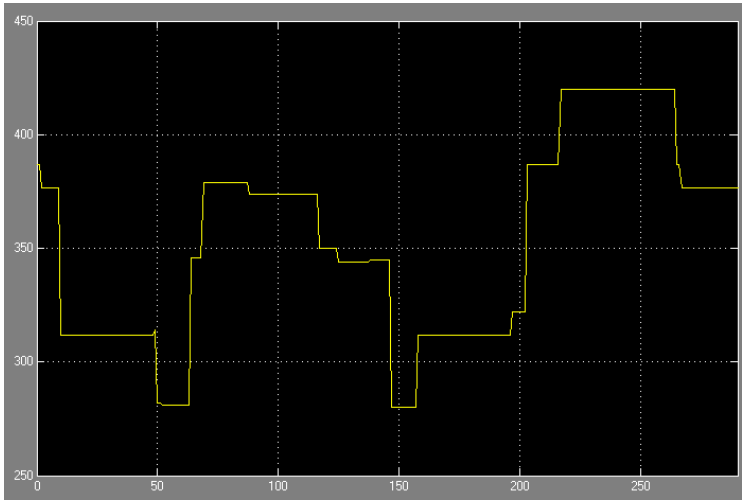

Input Set : Definitional Parameters
Steady State Short Circuit Current : A
Steady State Resistance : Ω
Open Circuit Voltage Annual Decline : Real
Resistance Annual Increase : Real
CurrentYear : Real
Reference to Power Bus : Power Bus
state variables : Power Bus SV Set
Reference Bus Voltage : Reference Bus Voltage
Reference Bus Voltage Value : V
Isc : A
Rrtgss : Ω
RrtgAD : Real
VocAD : Real
Vbus : V
Yr : Real
«constraint»
«placeholder.Model»
Degradation Model : MMRTG Degradation Model
DF : Real
MMRTG State Variables : MMRTG State Variables
MMRTG Current : MMRTG Current Output
Current Output Value : A
MMRTG Degradation : MMRTG Degradation Factor
Degradation Factor Value : Real
SteadyStateShortCircuitCurrent : A
BusVoltage : W
OutputCurrent : A
«placeholder.Model»
«constraint»
voltage-current model : MSL MMRTG 2011 Voltage-Current Curve
DegradationFactor : Real

JEO: SysML-Simulink Power Simulation





Individual Contributions

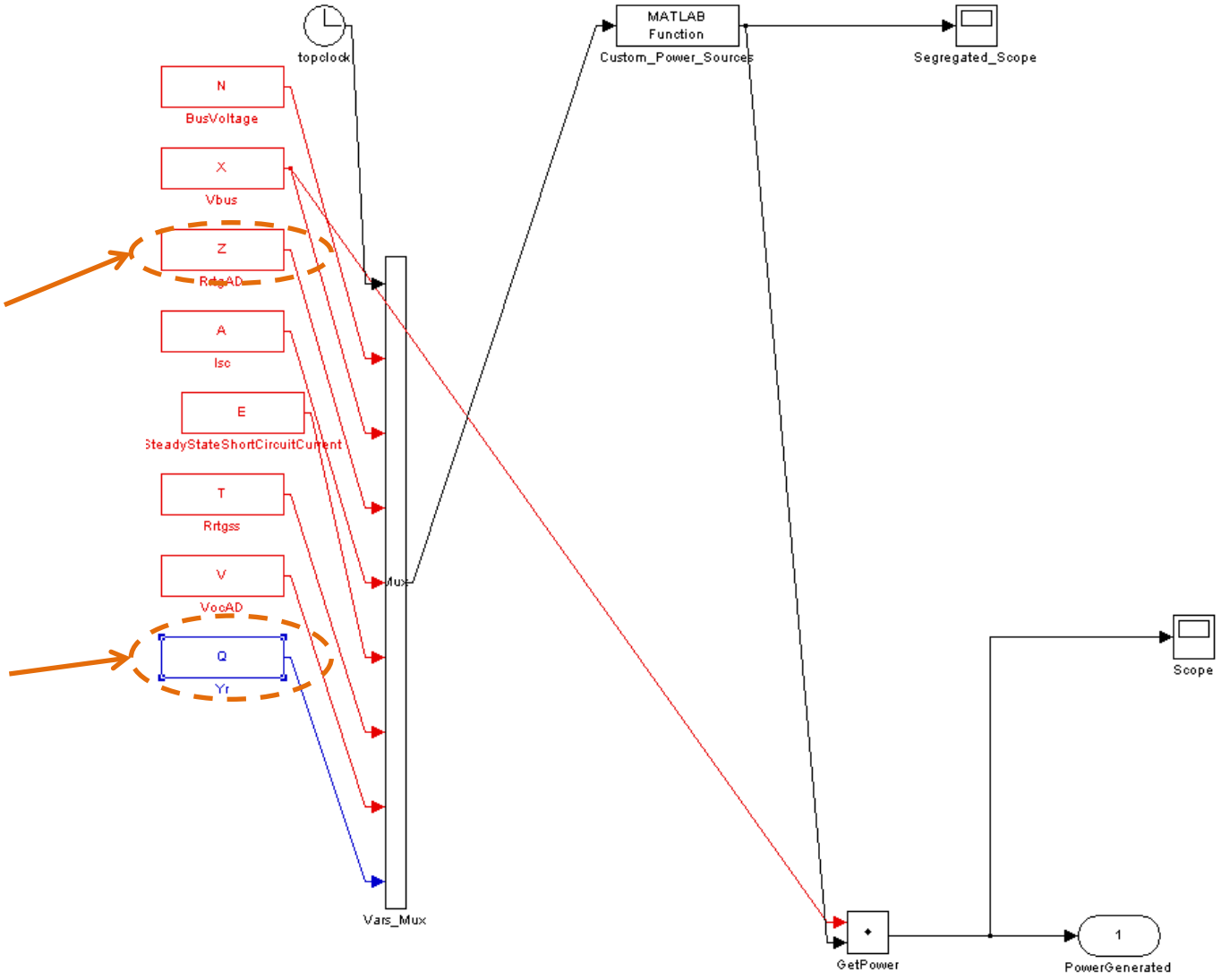


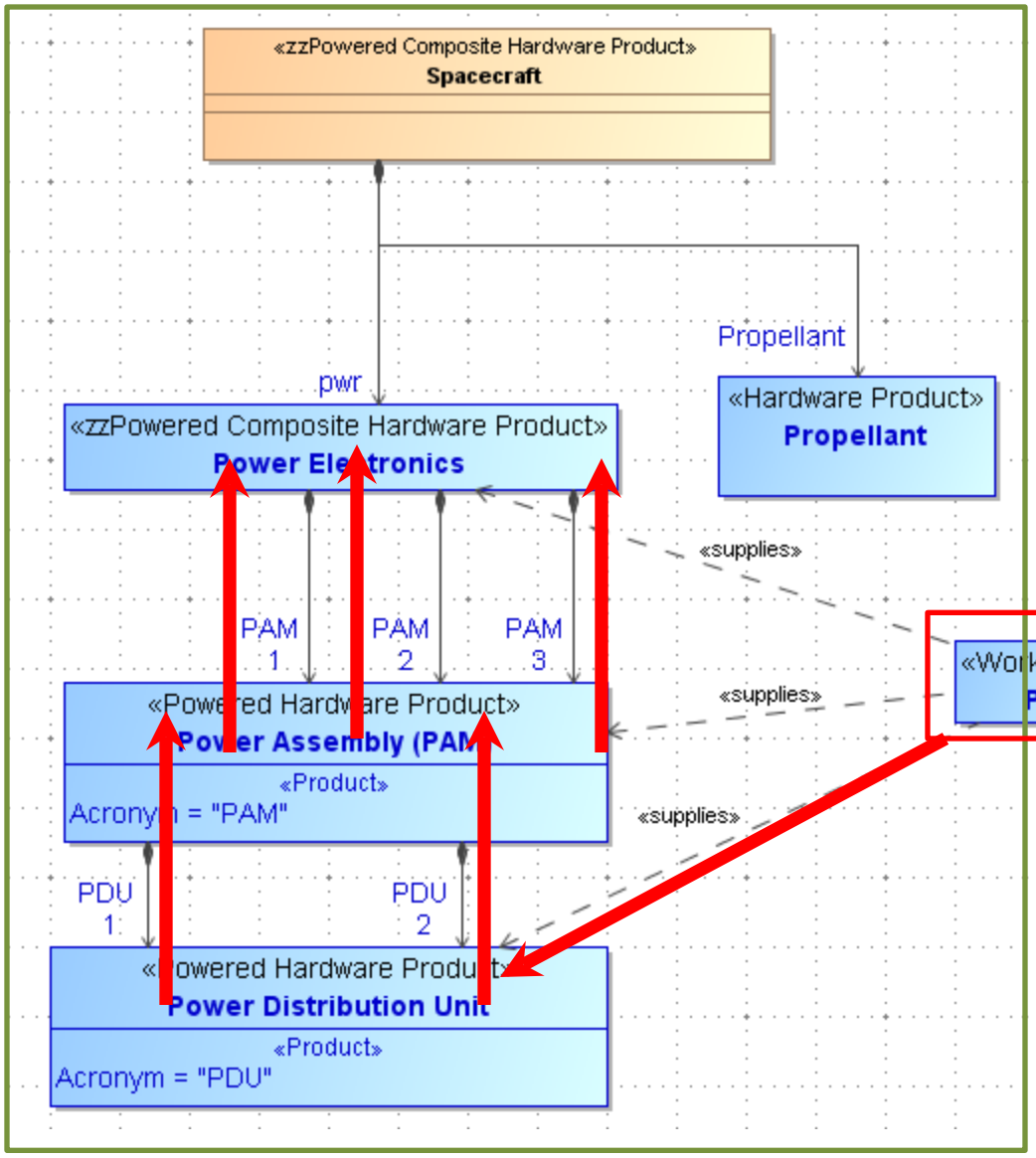
Power From Aggregate Loads

JEO: SysML-Simulink Power Simulation

Simulink Generation chain automatically identifies variables given values in the SysML model (treated as “givens”)…

…as well as those without values, that the user is encouraged to modify





Work MEL Rollup looks for leaves and then gets count by looking up composition tree (e.g, 6 PDUs in this case)

Composition MEL Rollup moves from top to bottom

What Does Transformation Buy?

- “Strong” transformation, where new models automatically spring entirely from older ones based only on definitions
- But “weak” transformation is still very useful, because it bootstraps information from old models into new ones
  - Synchronized by construction
  - Leverages redundancies for efficiency rather than getting bogged down in continuous synchronization and consistency efforts

Lessons Learned

- Good systems model is inherently multidisciplinary
  - Strong generalist with analytic skills
  - Specialist with discipline knowledge
- Requiring the systems engineer to provide data for execution by subsystem led to higher quality technical interactions than “give me your document”
- Computational sophistication runs into difficulties with human verification (how to “prove” that it is working) and validation
- It takes a lot of work to get a good system model – but much of this work is in *really* understanding the interfaces and approach (real systems engineering, not document shuffling)
- Be VERY careful to get your scope understood and locked in early or the model will run away from you!

Important Steps Forward

- Need to enlarge the QVTo community of practice if we intend to go forward with it as a technology
- Need to adapt transformation from a computer science “dark art” into systems-compatible thinking
- Start to categorize general classes of analyses to help with reusable transformation patterns
  - E.g., Geometric Finite-Element, Time Marching (ODE, Discrete Event), Parametric Web