National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

SysML-Modelica: A Redefinition & Modification Use Case

Alek Kerzhner and Nicolas Rouquette

Jet Propulsion Laboratory, Caltech

September 2012

Copyright 2011, California Institute of Technology
Government Sponsorship Acknowledged

Overview: Why are Modelica's redeclarations & modifications important for the SyM spec?

- Currently, Modelica's redeclarations/modifications are stored as strings
  - See: http://www.omg.org/spec/SyM/1.0/Beta3/
- Practically, this leads to 2 significant issues:
  - #1 (slide 14) Entering this information in text is error-prone
  - #2 (slide 17) Duplication is unavoidable

- JPL's proposal to capture Modelica's redeclarations/modifications
  - Cons:
    - Definitely an up-scope of SyM 1.0 Beta3
  - Pros:
    - The SyM 1.0 Beta3 transformations cannot support the use-cases described here
      - Because the SyM 1.0 Beta3 transformations are not designed to handle the duplication!

- Implications for the SyM transformations (slides 20,21)
  - Separate concerns that were previously entangled in SyM 1.0 Beta3:
    - Modelica's resolution semantics (see Chapter 4 scoping, name lookup & flattening)
    - SysML (with UML-based specialization) ⇔ Resolved Modelica Mappings

Use Case Overview

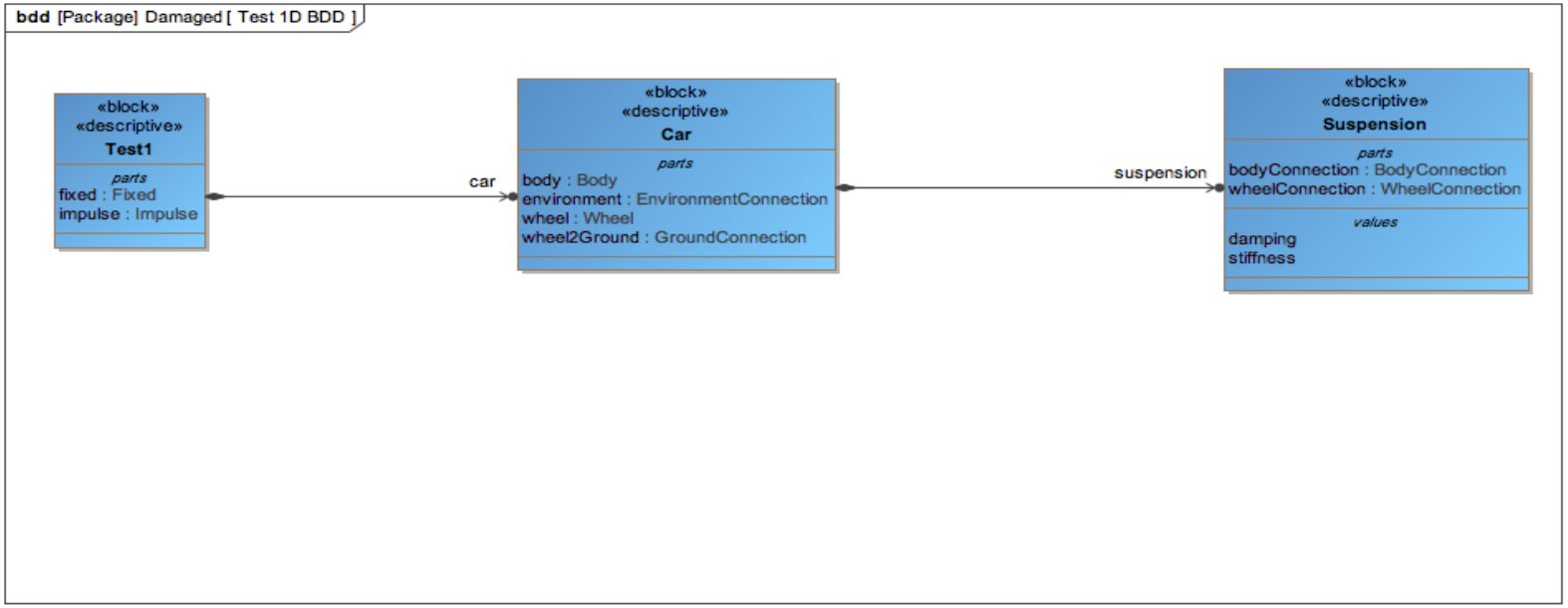
- We want to be able to have a set of “canned” analyses which can be used as is or modified by experienced users:
  - Analyses would be predefined in a Modelica tool where it is more natural to deal with equations, etc..
  - Analyses would then be imported into SysML using the M2S transformation
  - Users would connect these analyses as-is to descriptive model elements
  - Users would use SysML generation/redefinition to modify the analysis models
  - The S2M transformation would then create the appropriate Modelica model
- Why?
  - We cannot anticipate all variations of analyses of complex models a priori
  - We need to facilitate exploring different ways to analyze an existing model (and compare them!)
- This could support:
  - Analyzing different system variants using the same base analysis
    - Could use redefinition to replace the analysis models used for one or more components
  - Analyzing the same system during different lifecycle phases
    - Could use redefinition to adjust the equations describing broken/damaged components

Use Case Overview (cont'd)

We want to perform two tests on a car suspension:

- when the suspension is healthy (as designed)
- when it has been damaged

We start from the as-designed test (analysis models exist in Modelica, have been imported into SysML):



National Aeronautics and
Space Administration

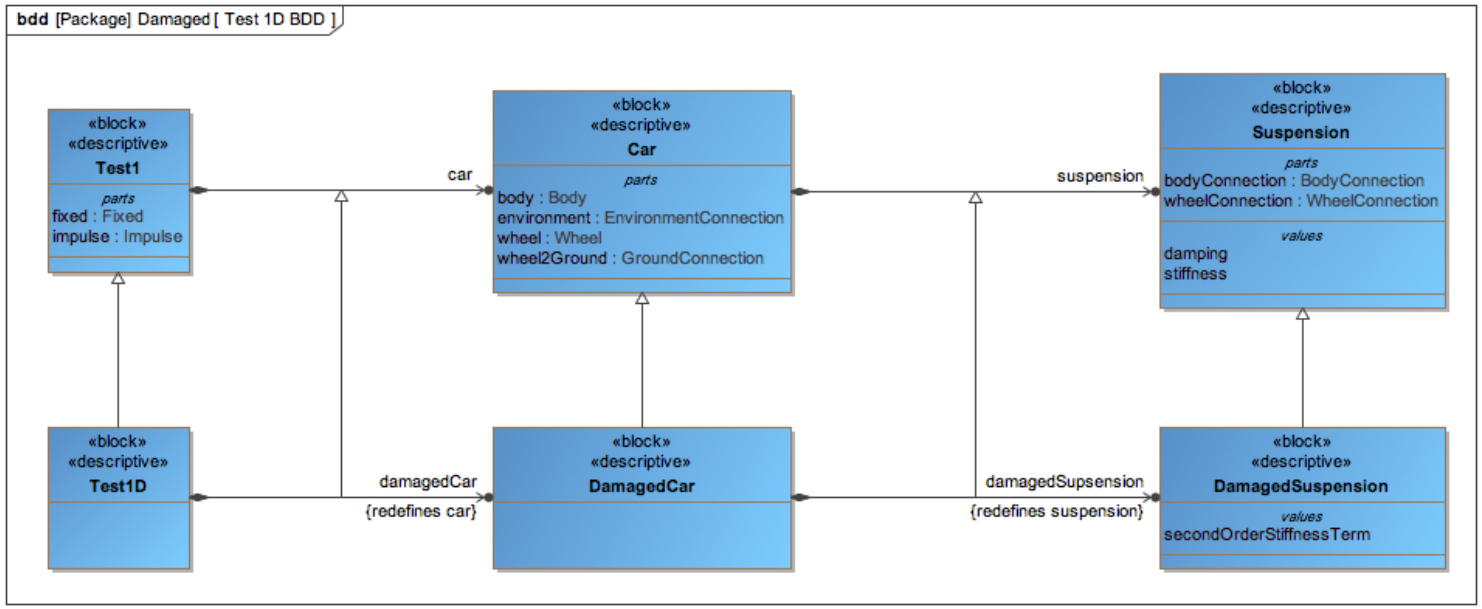
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Use Case Overview (cont'd)

We want to perform two tests on a car suspension:

- when the suspension is healthy (as designed)
- when it has been damaged

On the descriptive side, we can use SysML specializations to define the test for the damaged suspension:

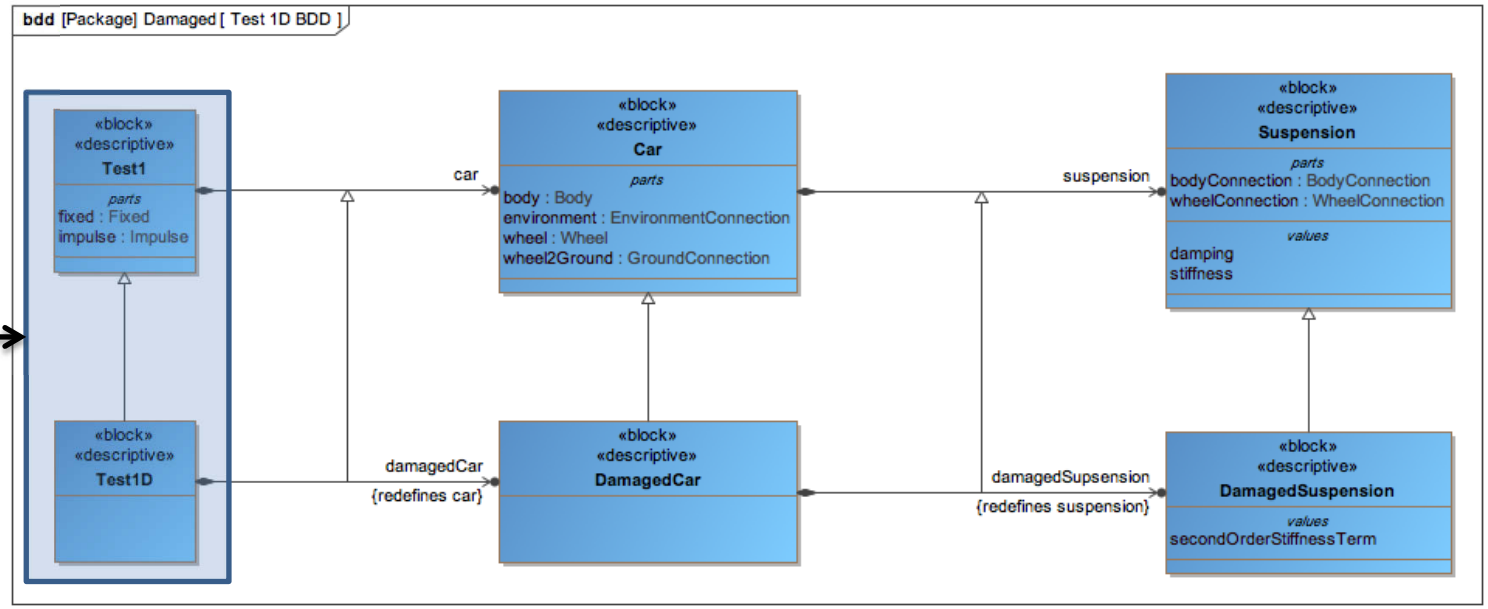


Use Case Overview (cont'd)

We want to perform two tests on a car suspension:

- when the suspension is healthy (as designed)
- when it has been damaged

We want to simulate these tests



National Aeronautics and
Space Administration

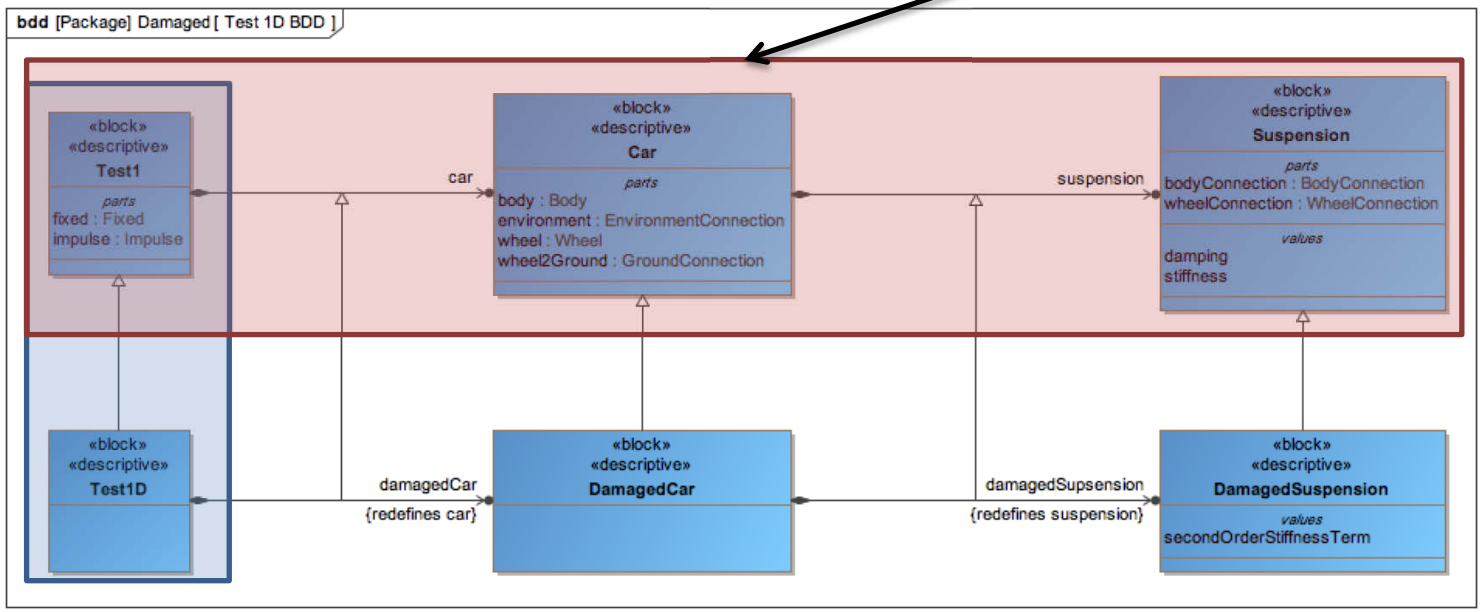
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Use Case Overview (cont'd)

We want to perform two tests on a car suspension:

- when the suspension is healthy (as designed)
- when it has been damaged

Analysis Models for these already exist in Modelica



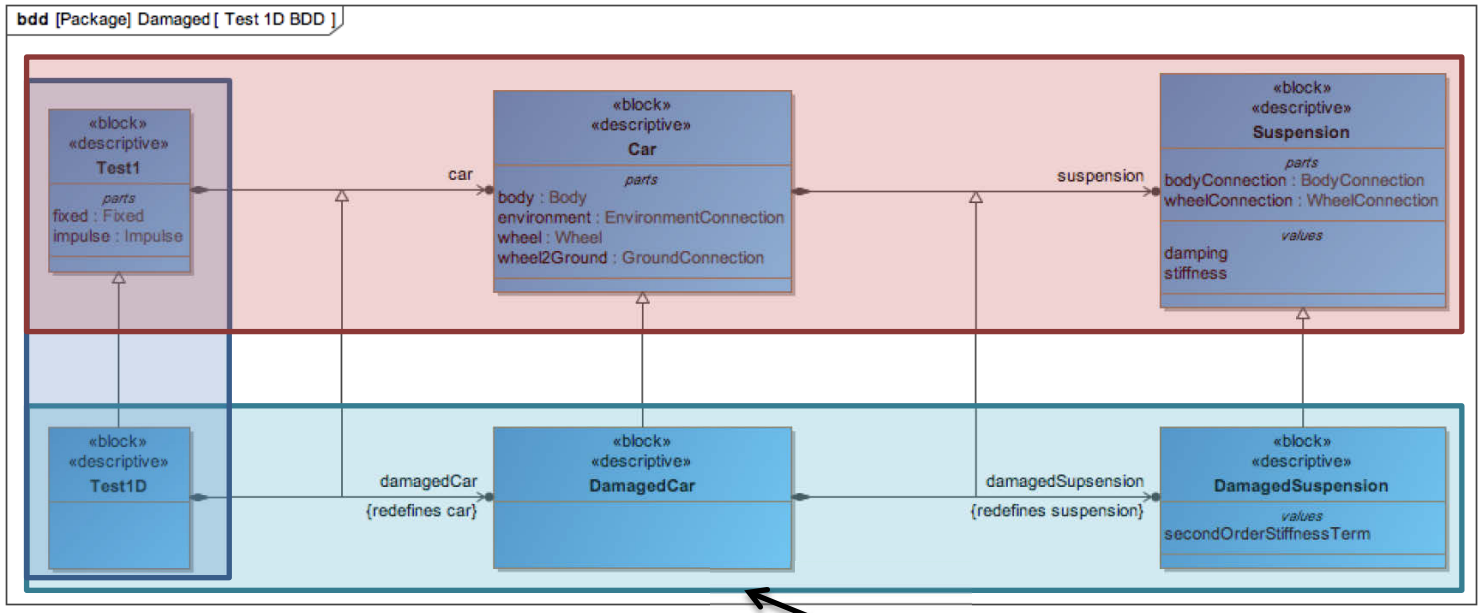
National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Use Case Overview (cont'd)

We want to perform two tests on a car suspension:

- when the suspension is healthy (as designed)
- when it has been damaged



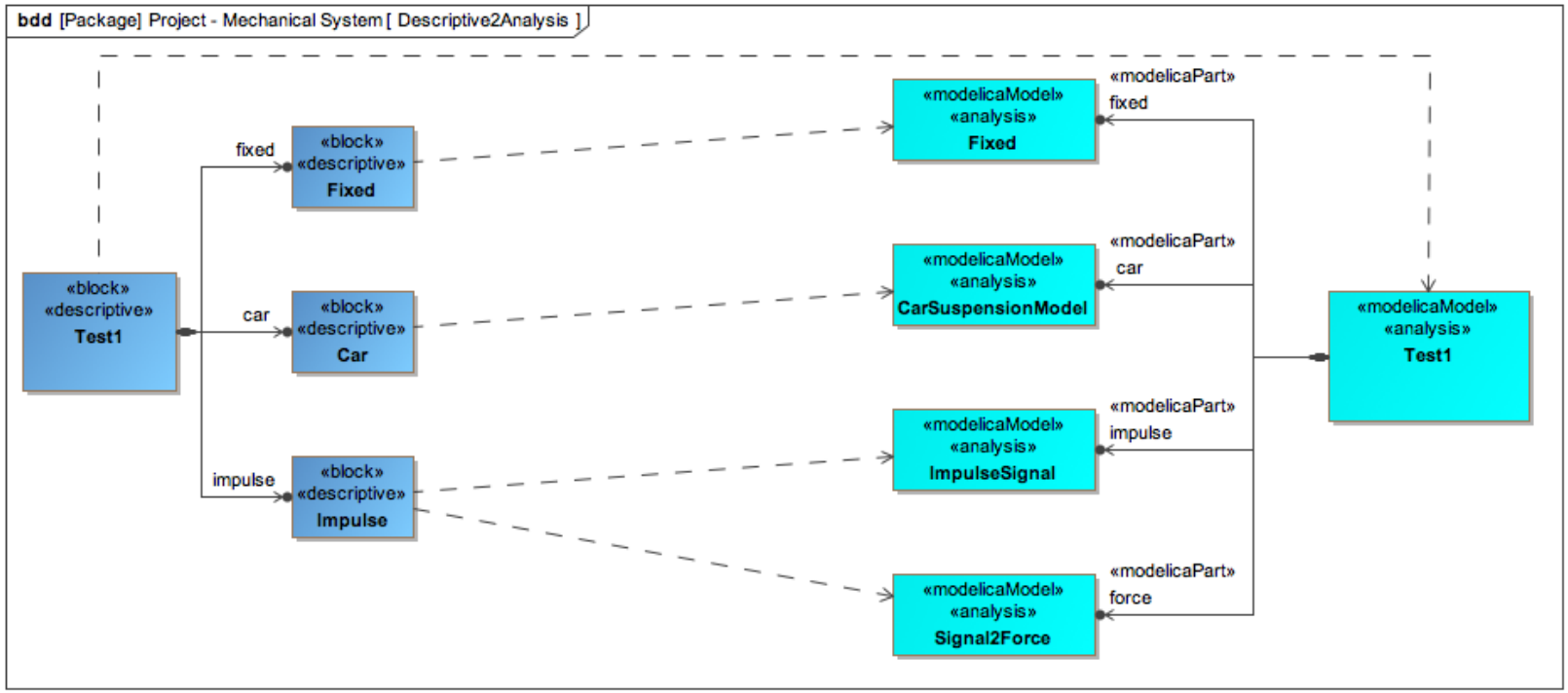
Analysis Models for these need to be created

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Relationship between descriptive and analytical models

Let's assume for part of the descriptive model, there exist appropriate predefined analyses

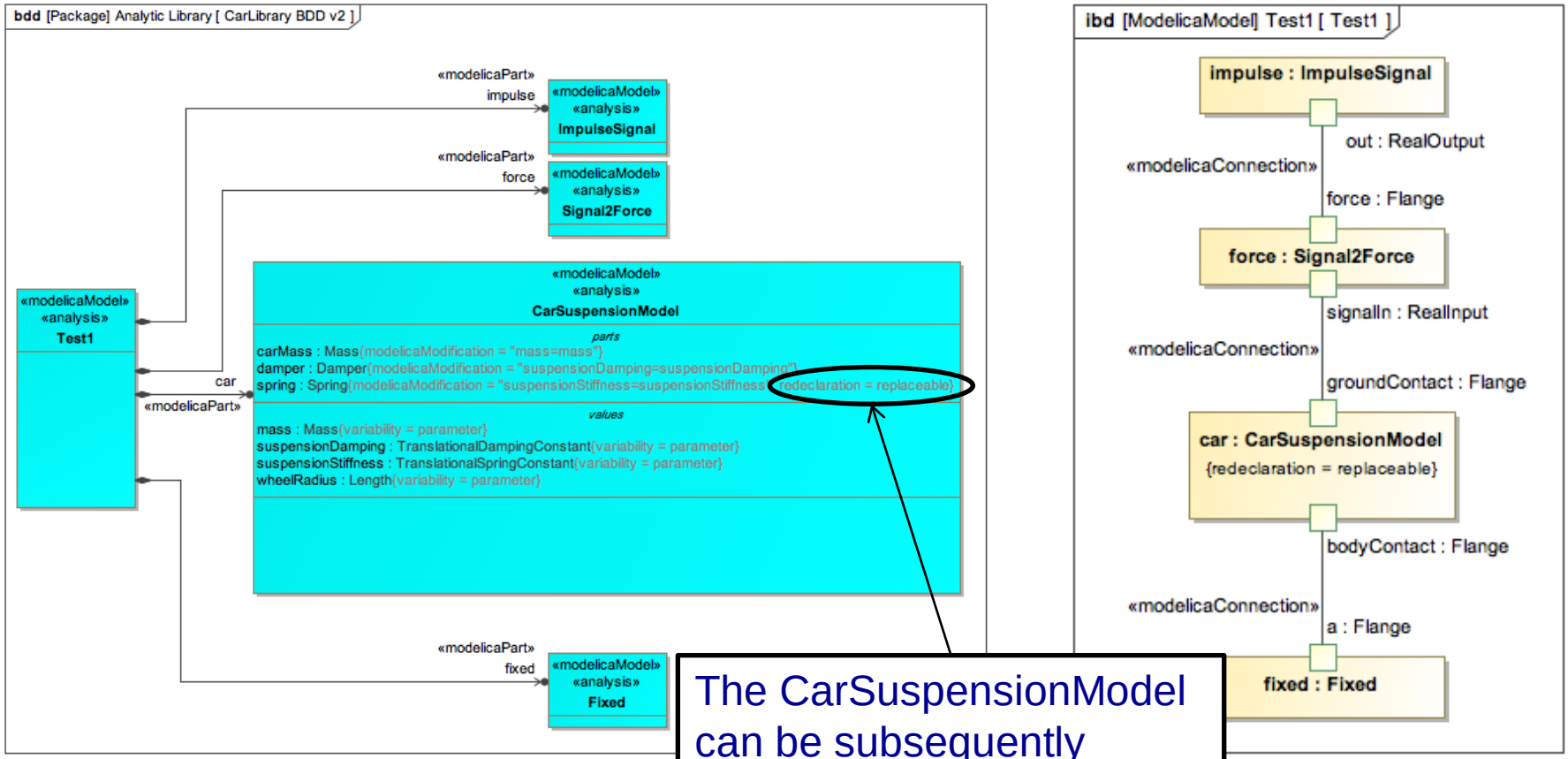


National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Existing Test Model

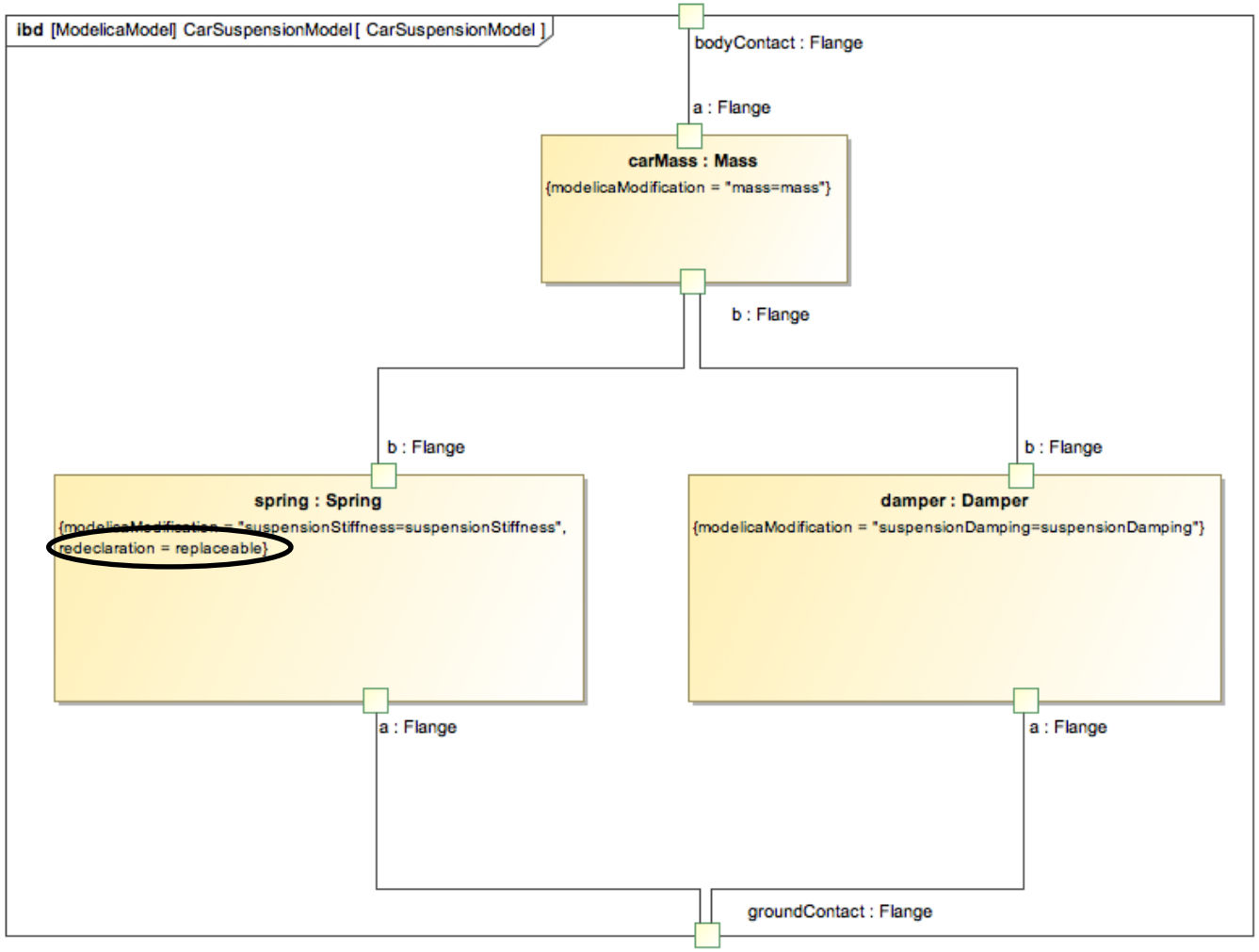
This includes the existing test model, which contains a Car Suspension model that can be redefined (See SyM Beta 3, Section 10.3 p. 26) . Specializing CarSuspensionModel is precisely what we need to define the damaged car variant for the 2nd test.



The CarSuspensionModel can be subsequently modified (by specialization)

Existing Car Model

The internals of the car suspension model:



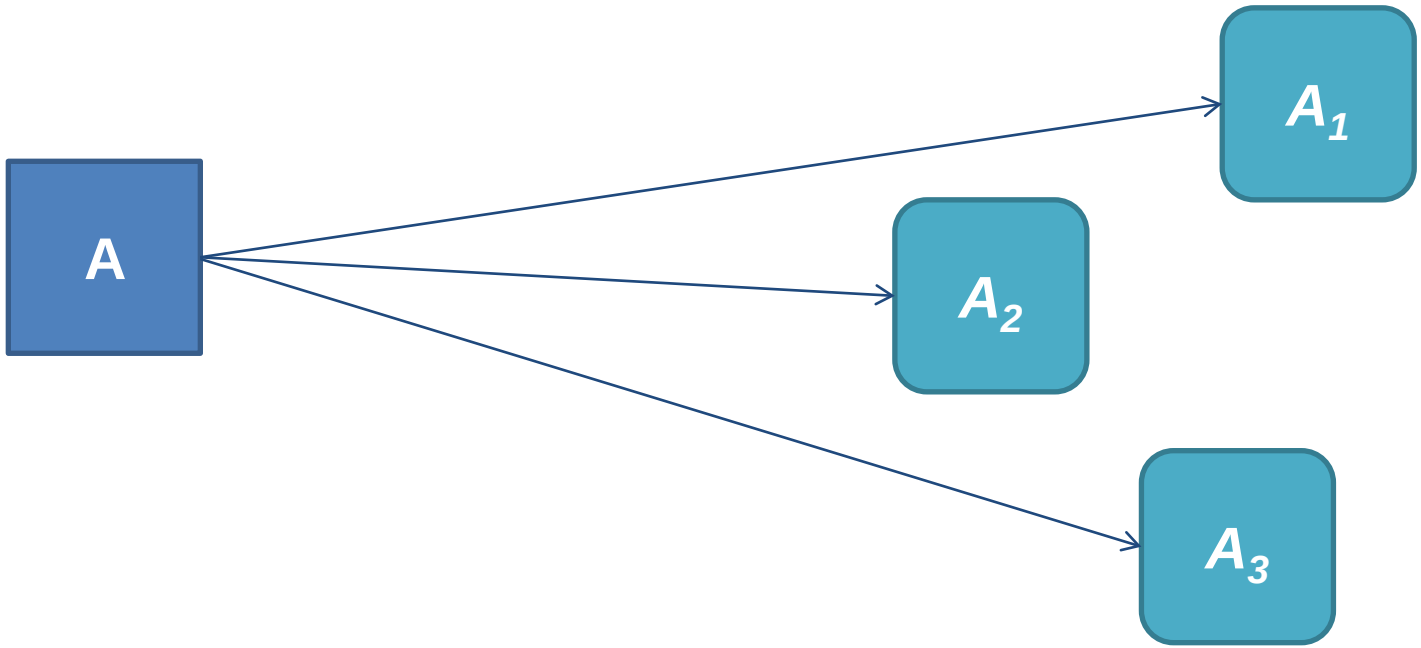
Modelica's replaceable annotations are very important.

Without them, we couldn't reuse this model to do the damaged suspension test.

(See SyM Beta 3, Section 10.3 p. 26)

Analysis Contexts

The same descriptive model… …could be analyzed in various ways



- Analysis contexts capture the correspondence between a descriptive model and a particular analysis. (See SyM 1.0 Beta3, p. 76)
- Analysis contexts are one example where cross-cutting relationships exist between an analysis and the rest of the descriptive model.

National Aeronautics and
Space Administration

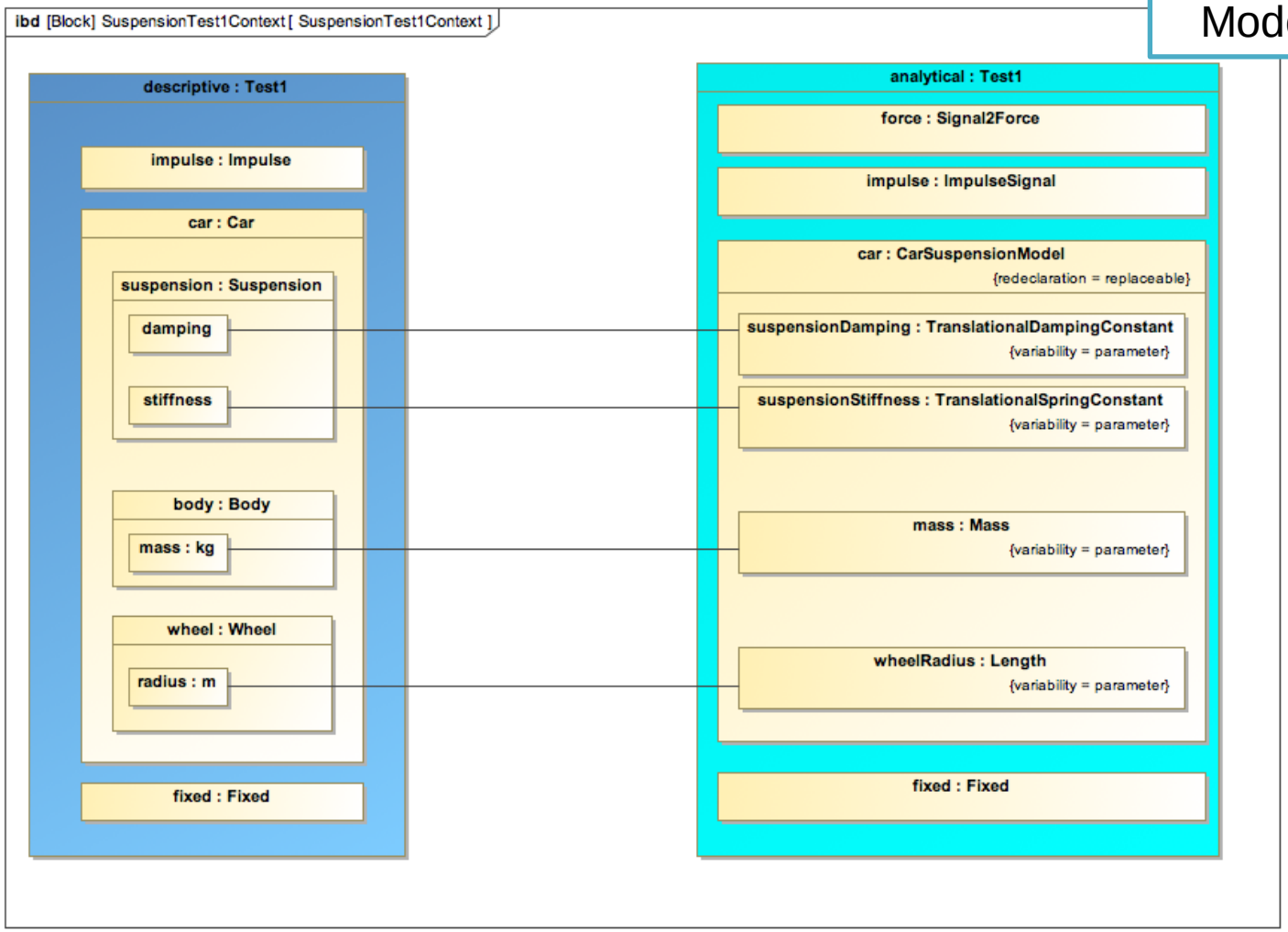
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Analysis Context for Original Test

For the original test, one can capture the correspondence between a structural/descriptive model and the analysis model

Descriptive: SysML

Analytical: Modelica

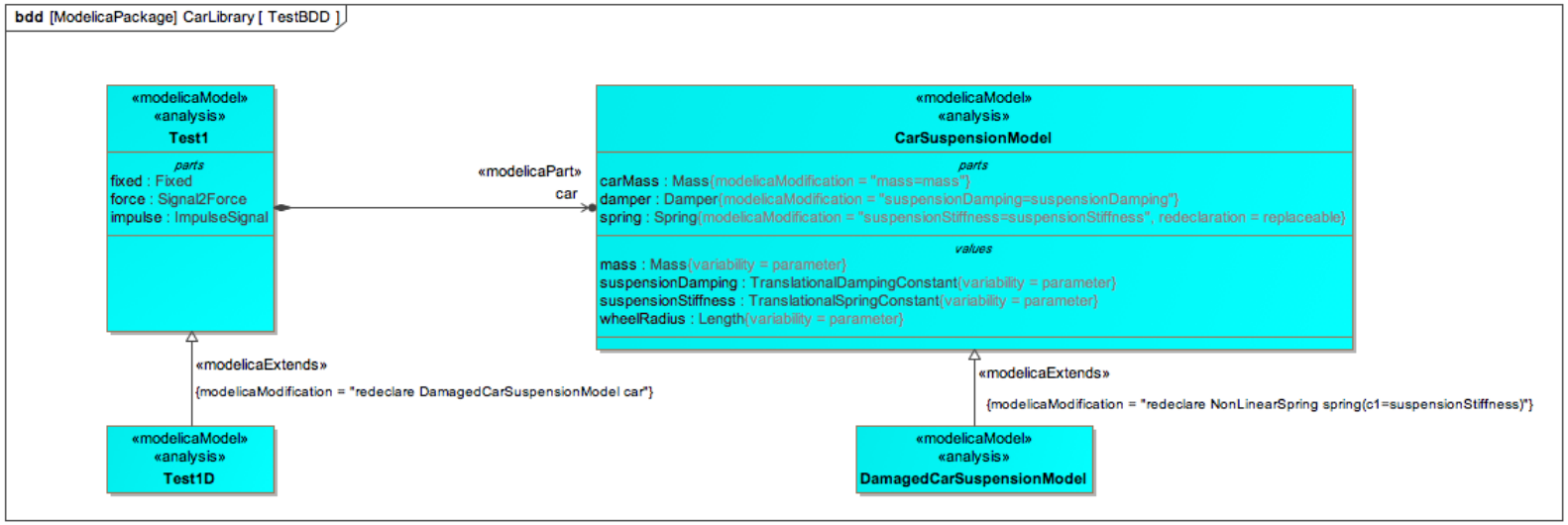


(See SyM beta 3 p. 67, 75-76.)

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Defining the new models with the current specification



(See SyM Beta 1.3, Section 8.10, p. 17)

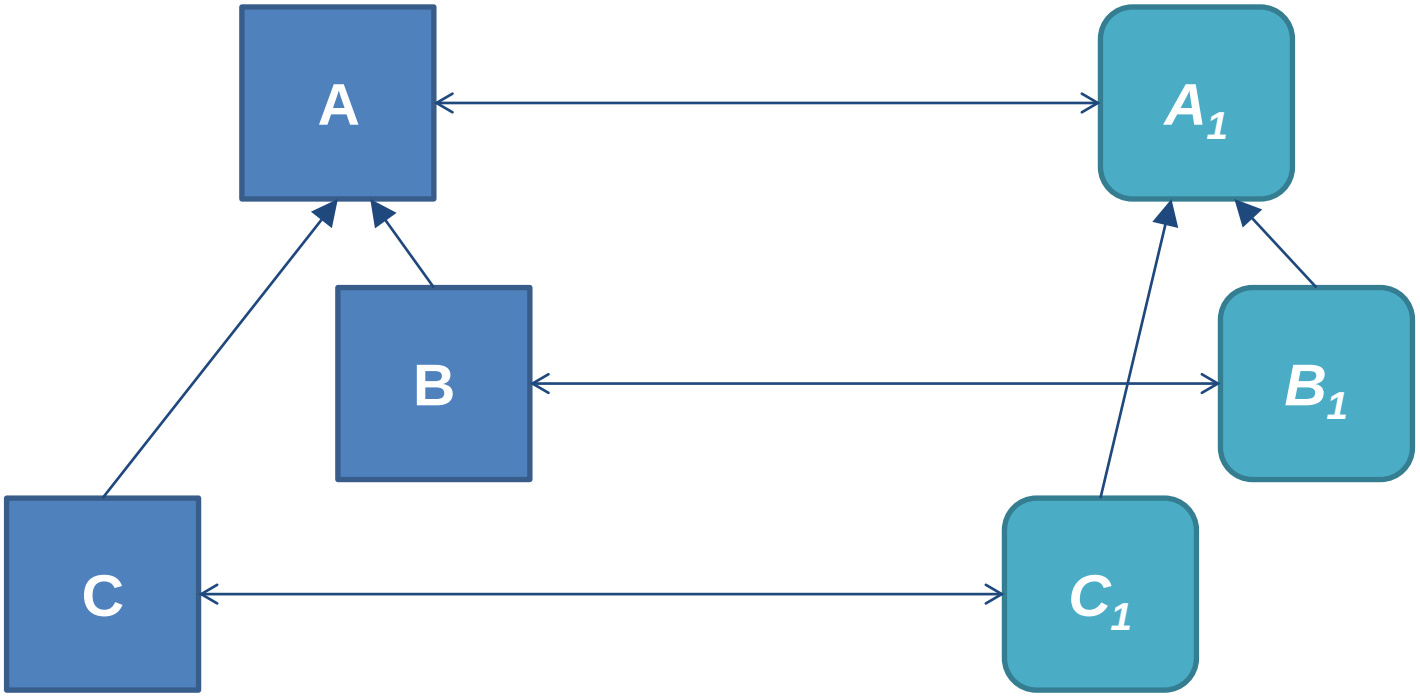
Issues (1/2):

#1 The user needs to manually enter and interpret the information

- There is no way to insure consistency with the model or perform checks on these modifications until compile time
- Not only does the user now need to know a new syntax, they also need to avoid typos and understand the structure of the underlying models
- Understand the appropriate name lookup and namespace (mass=mass)

Variant Analysis Contexts

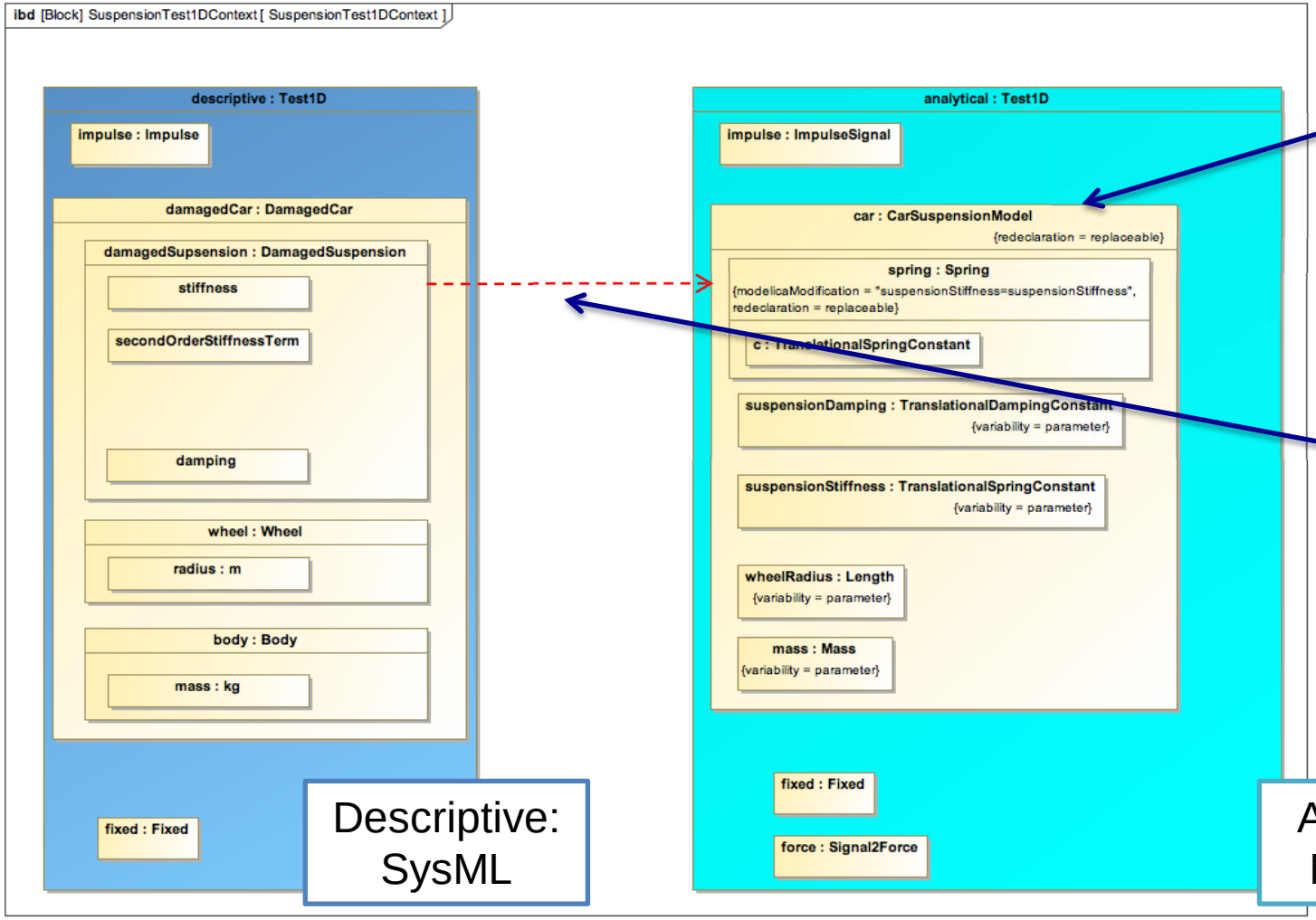
Variations of a descriptive model… …could be analyzed in various ways



- For each descriptive variant, an analysis context is also needed

Analysis Context

With only the tag value strings, the correspondences cannot be captured because the appropriate structural elements are missing



This type has been re-declared, but that is only captured in the text.

We can't show the descriptive/analytical correspondences (redeclaration is only textual!)

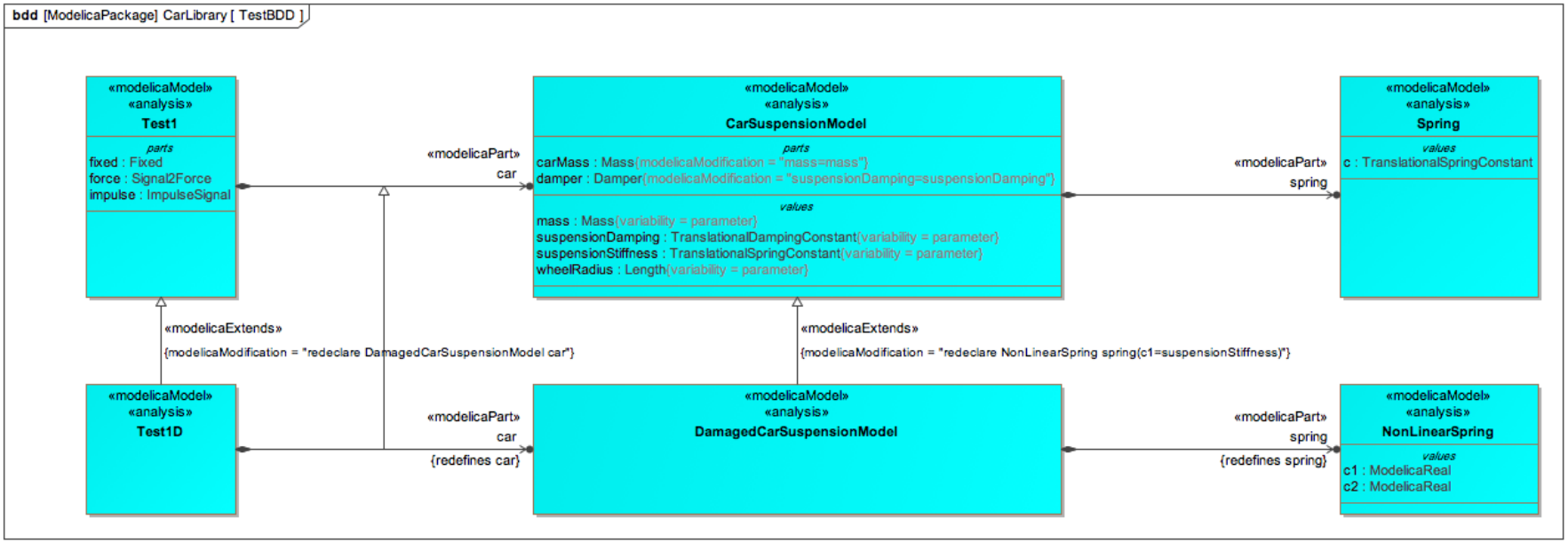
Descriptive: SysML

Analytical: Modelica

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Defining the new models with the current specification



Issues (2/2):

#2: In order to support the definition of references to other parts of the model, the user needs to create the redefinitions anyway!

National Aeronautics and
Space Administration

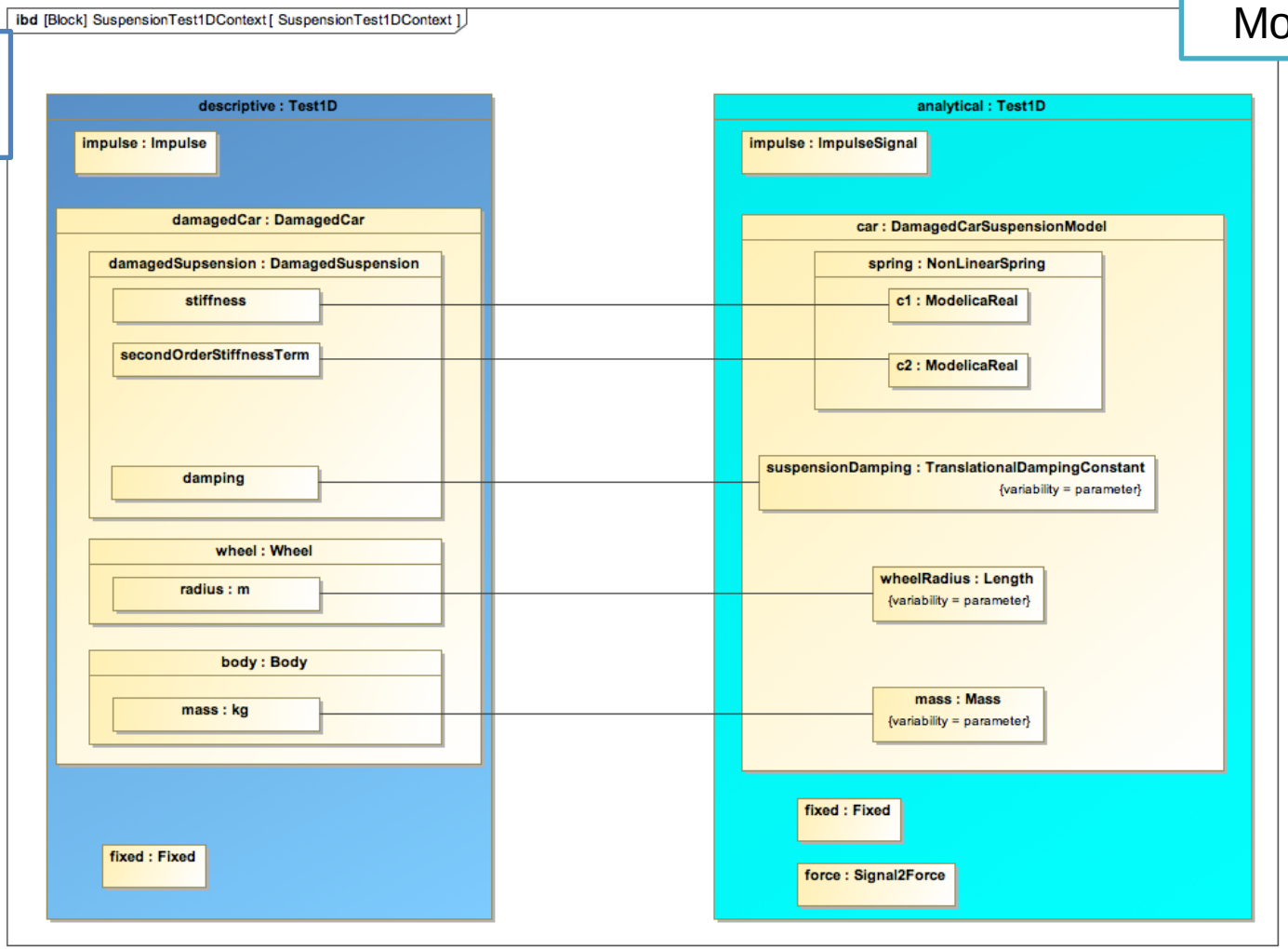
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

New Analysis Context

Now the correspondences can be captured because a redefined car property is available.

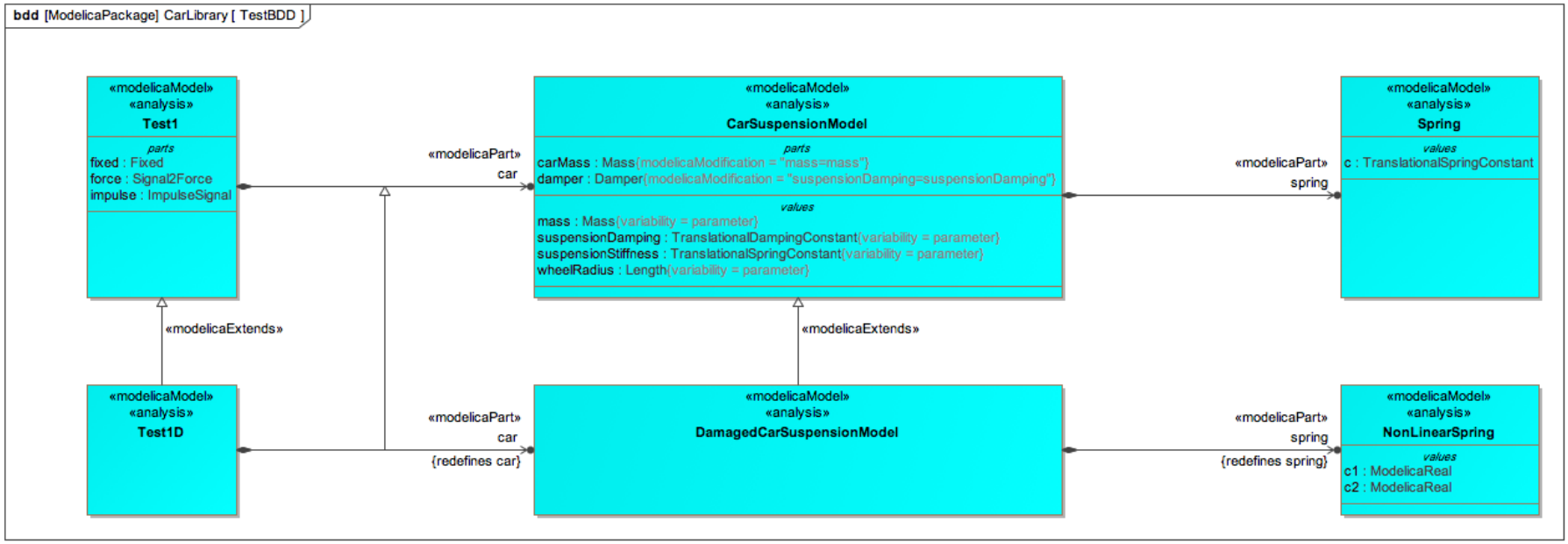
Descriptive: SysML

Analytical: Modelica

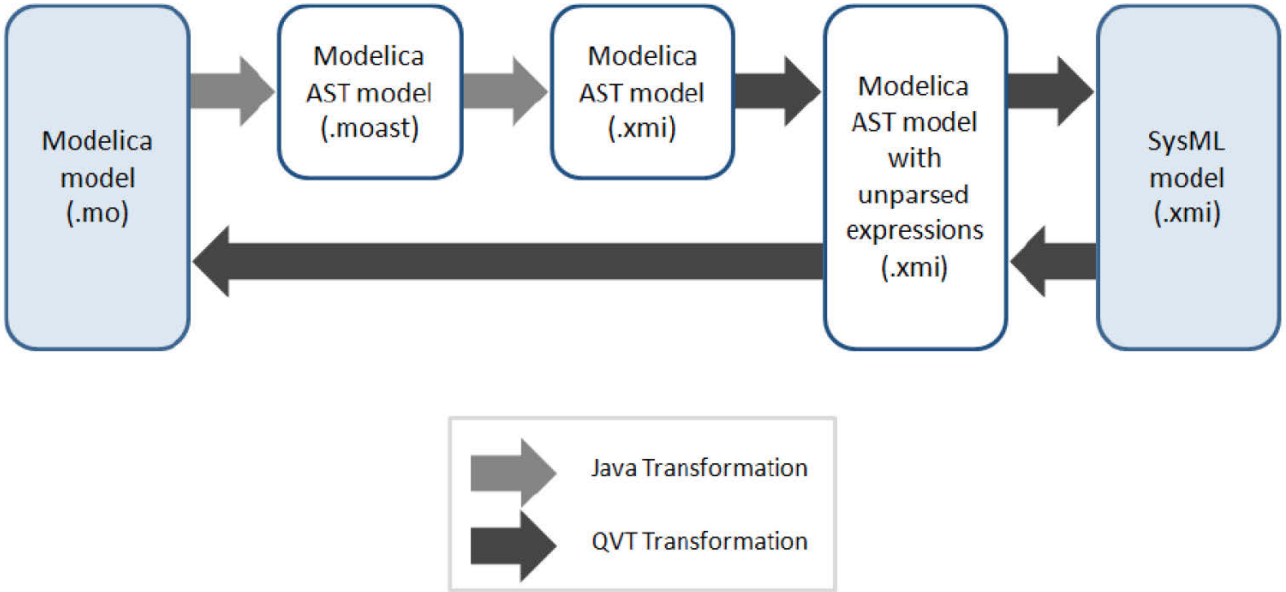


Proposed Solution

- Avoid duplicating the representation of Modelica's redeclaration as tag value strings and UML/SysML's redefinitions
- Instead, specify the correspondence between:
  - Modelica redeclaration
  - UML/SysML redefinition (i.e., the canonical representation of Modelica redeclaration in UML/SysML)



Impact on the SysML/Modelica Transformations: Currently



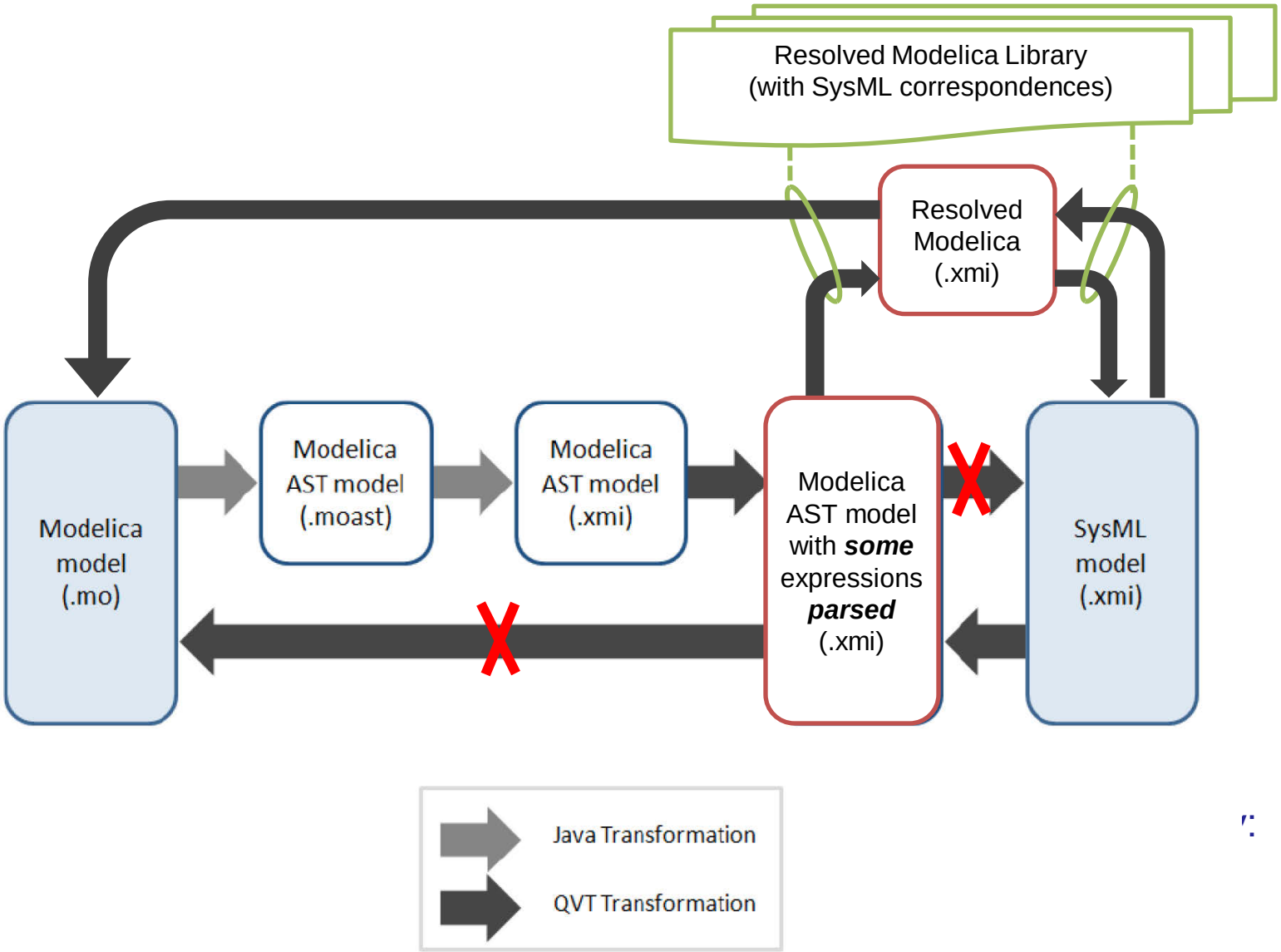
The Modelica AST ⇔ SysML transformations entangle two aspects in a fragile way:

- Modelica's resolution semantics (See Modelica Chapter 4)
  - A Modelica's class component is "looked up" by name in the corresponding SysML Block
- The mapping between Modelica & SysML is fragile
  - Modelica's redeclarations/modifications are stored as strings (Modelica => SysML)
  - Errors due to unexpected duplication between String-based Modelica redeclarations/modifications and UML-based specialization (SysML => Modelica)

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Impact on the SysML/Modelica Transformations: Proposed



Modelica Resolution & Modelica/SysML Correspondences

- Modelica Resolver
  - Currently, a QVT transformation: Modelica => Modelica
    - Authoritative reference is Chapter 5 of the Modelica Language Reference
  - Modelica Metamodel must be isomorphic to the record-based representation produce by a Modelica compiler implementation (flattener)
    - There is currently no agreement among implementation about what this record-based representation
    - This makes the implementation of a .mo ⇔ .xmi (modelica) converter highly dependent on a particular Modelica compiler implementation
  - Augments the SyM 1.0 Beta3 Modelica metamodel
    - Adds attributes to represent references to resolved elements
      - Example: ComponentRef needs a reference to a Component definition

      (see SyM 1.0 Beta3 section 14.2.12)

- Modelica ⇔ SysML Correspondences
  - Modelica's redeclarations & modifications are *relationships,* (not elements)
  - To map these relationships to SysML, we need to incrementally maintain element/element correspondences between Modelica & SysML