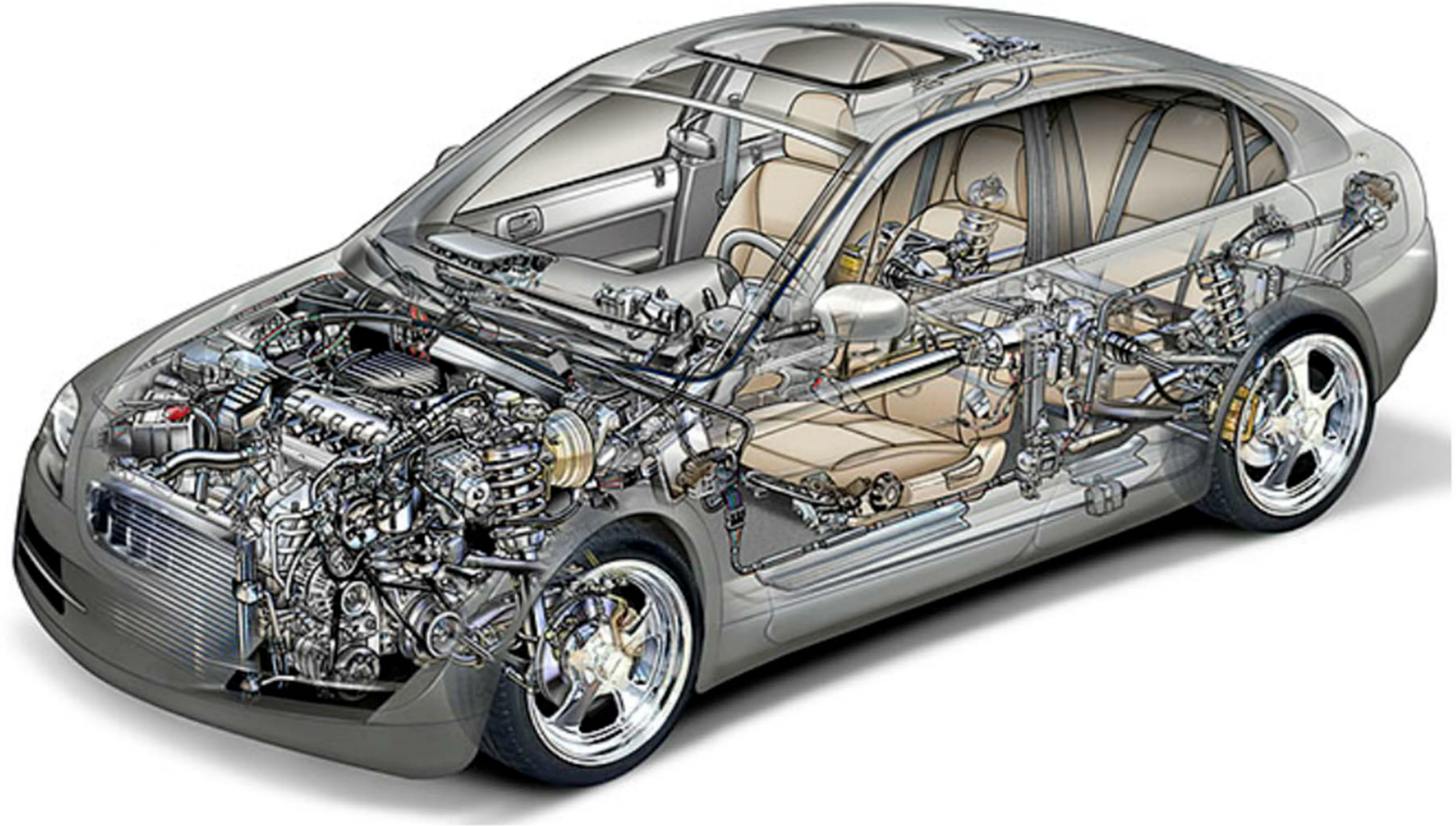
QUADAI ARCHITECTURE DERIVATION



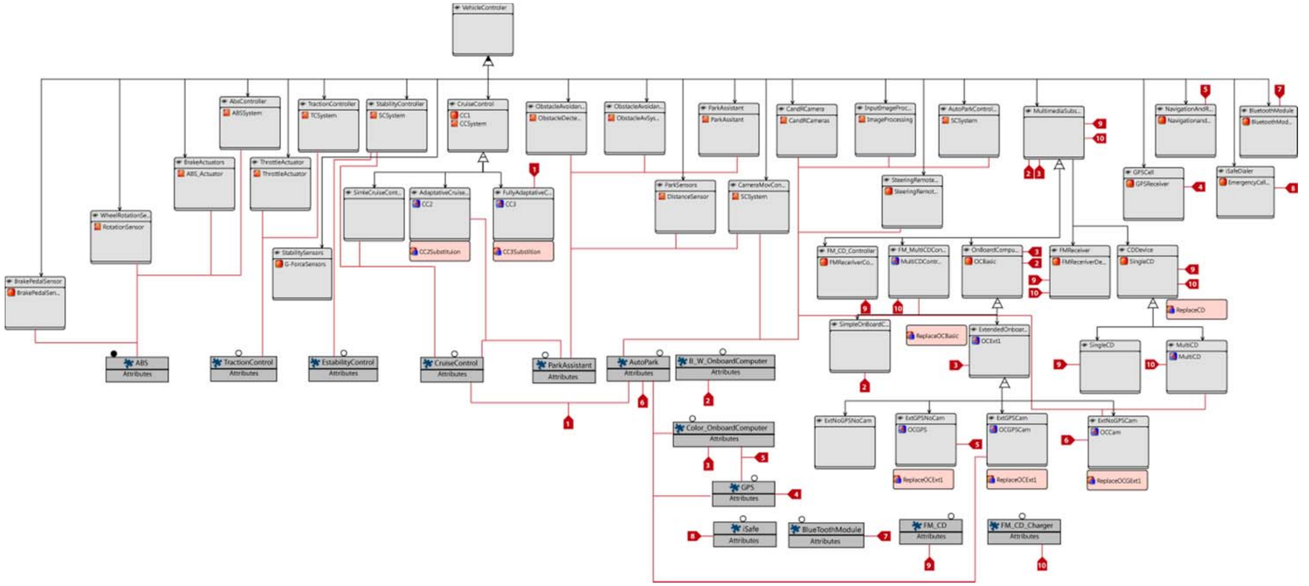
CarCar Case Study

Introduction

- The software product lines (SPL) approach emerged with the aim of improving software development processes so as to reduce costs and substantially enhance productivity and product quality.
- In SPL development the software architecture plays a key role.
  - In conventional software development processes the software architectures are the means of attaining the required quality attributes.
  - When applying the SPL approach this effect is even more pronounced. We depend on a product line architecture (PLA) that supports the possible configurations and on a derivation process for obtaining the product architecture of the product under development as an instance of the PLA.
- In SPL development the PLA includes a set of variation mechanisms to support the variability among products.

Problem Statement

- The product configuration and architecture derivation is a complex, time-consuming process in witch different aspects should be taken into account.
- Given a set of architectural variation points: how to decide which ones should be selected or which others should not?



- For solving that problem we have proposed QuaDAI

Introduction to QuaDAI

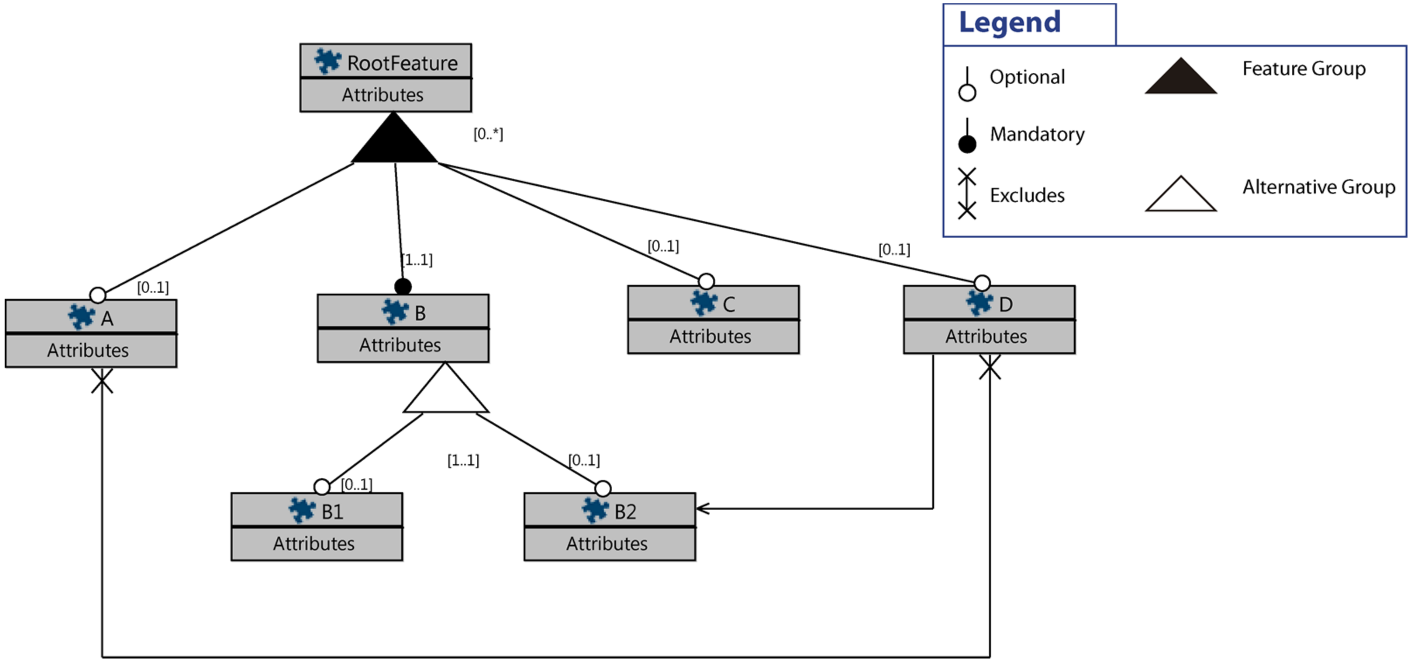
- QuaDAI is a integrated method for the derivation, evaluation and improvement of software architectures
  - Relies on an artefact (a multimodel) in which we can represent the different viewpoints of the SPL can be defined (variability architecture, quality), and we can establish relationships among elements on these viewpoints.
  - The Multimodel represents the SPL domain engineering information and it will be used in the application engineering step.
  - The relationships among the architectural viewpoint, the variability viewpoint and the quality viewpoint are used during the product configuration & product architecture derivation stage to:
    - Help in the selection features, non-functional requirements and quality attributes to be included in the configuration
    - Resolve the Product line architectural variability during the product architecture derivation process.

The multimodel in product configuration and architecture derivation

- The multimodel for product configuration and architecture derivation is composed of three viewpoints
  - **The variability viewpoint:** containing the commonalities and variability within the product line. Its main element is the feature, which is a user-visible aspect or characteristic of a system
  - **The architectural:** containing the architectural variability of the PL Architecture
  - **The quality viewpoint:** containing the quality attributes, and the NFRs for both the SPL and the product under development

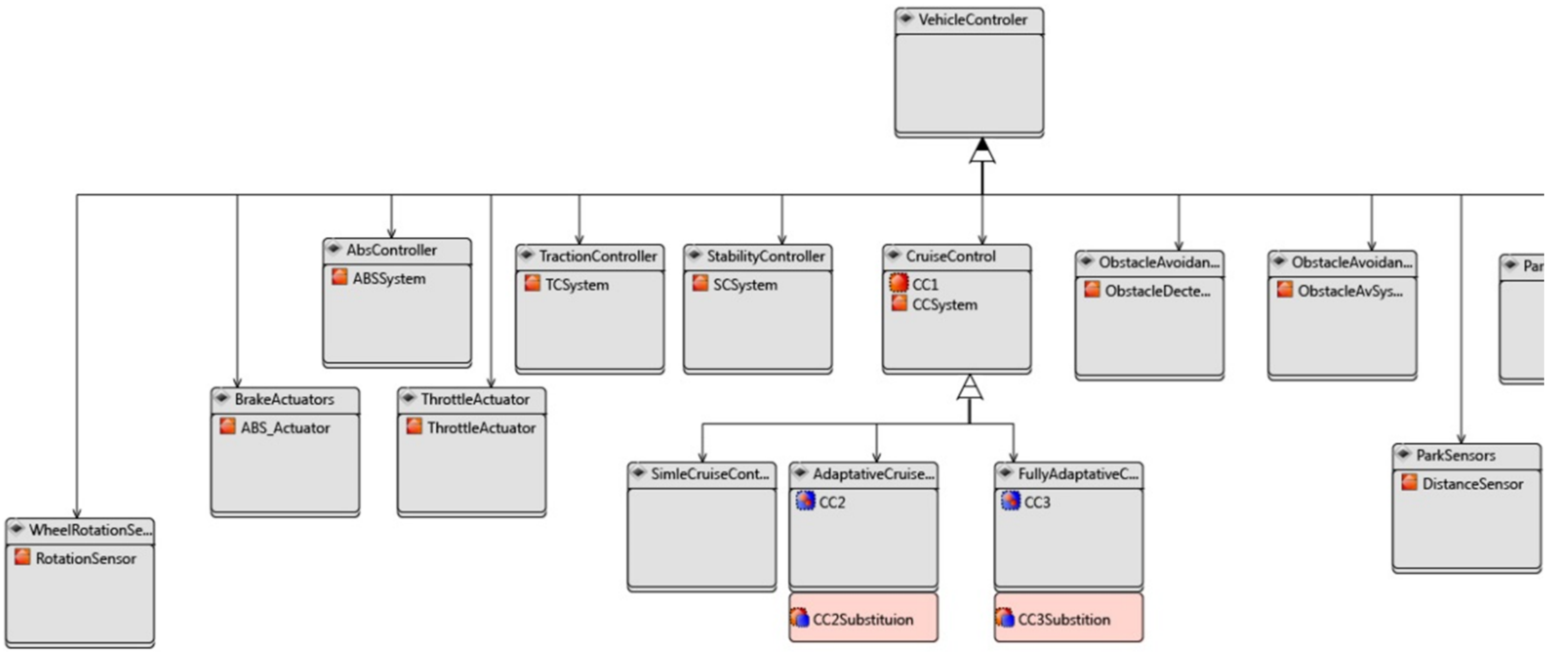
The variability viewpoint

- For defining the variability viewpoint of the multimodel we use a cardinality base feature model.



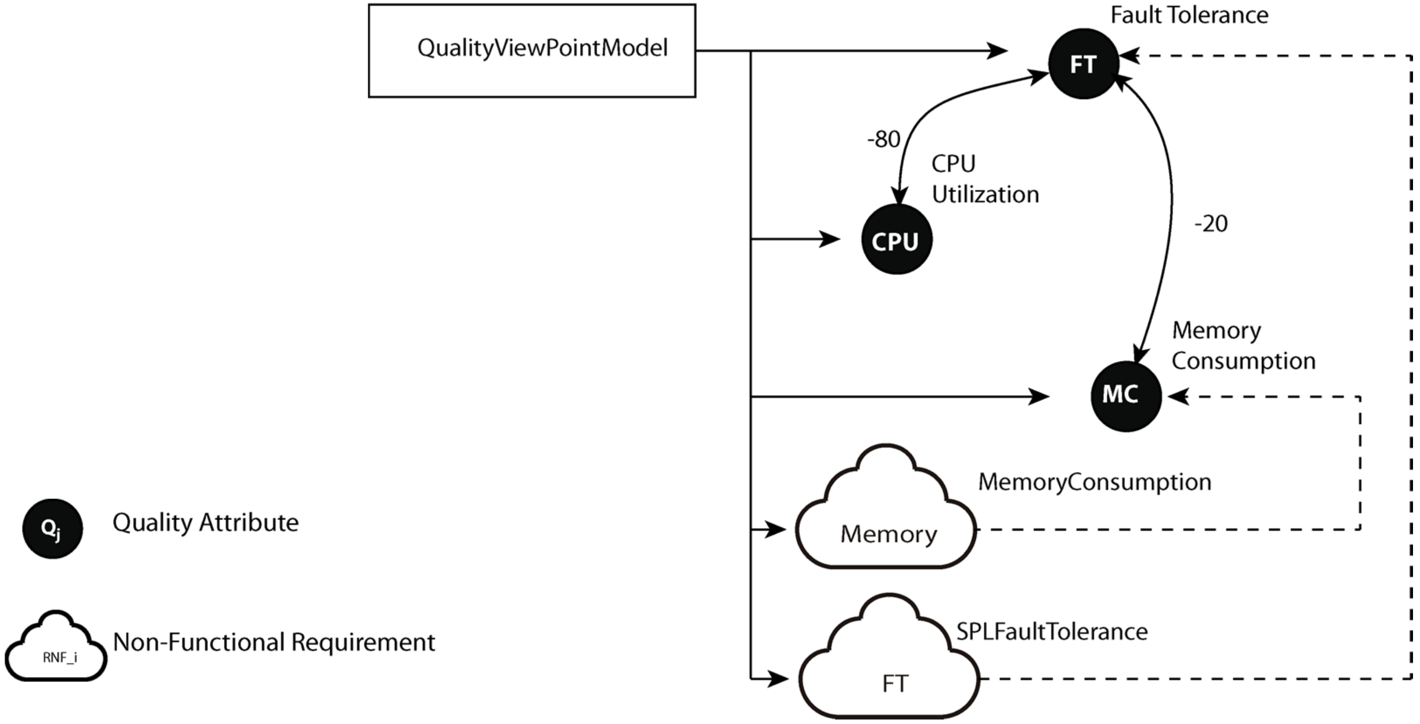
Architectual Viewpoint

- For expressing the variability in the PLA we use the Common Variability Languaje (OMG Standard)



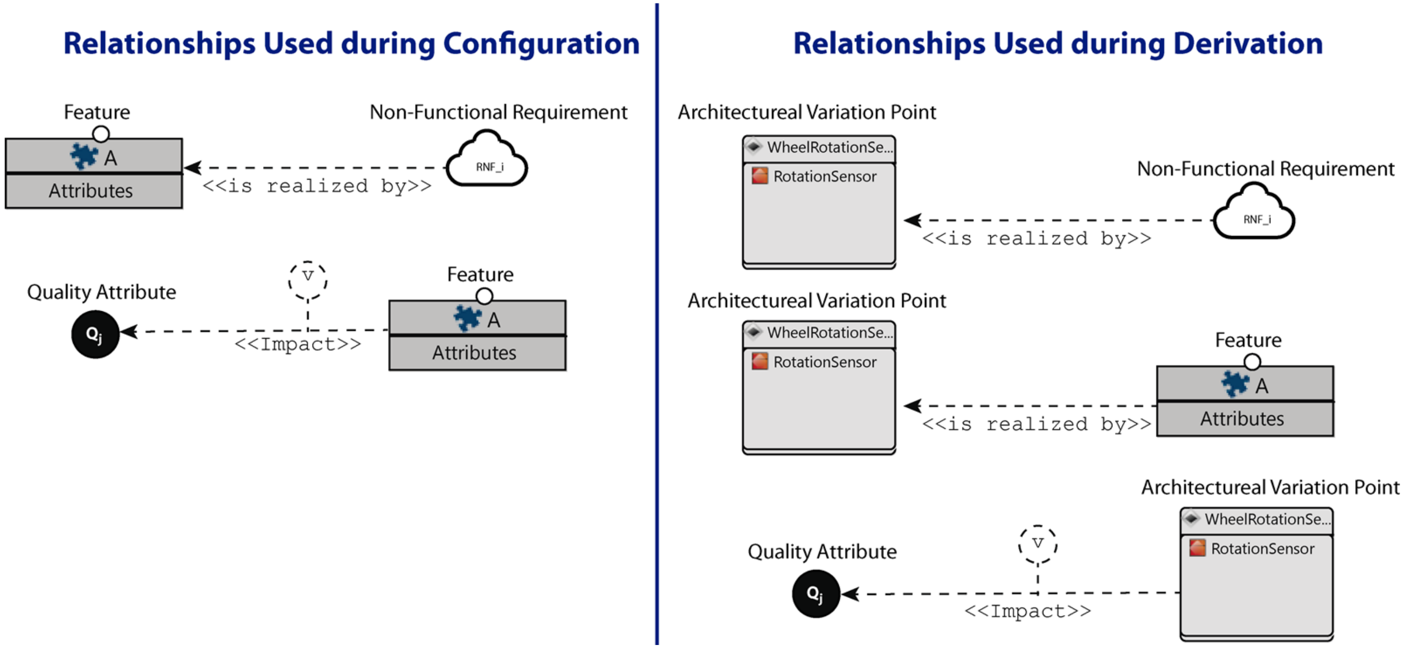
Quality Viewpoint

- For defining the Quality Viewpoint we use a Quality Model for SPLs where we can:
  - Define the impact relationships between quality attributes
  - Define the NFRs for both the SPL and the specific products
  - During the configuration, we can both select the NFRs and prioritize the quality attributes of a give product



Relationships

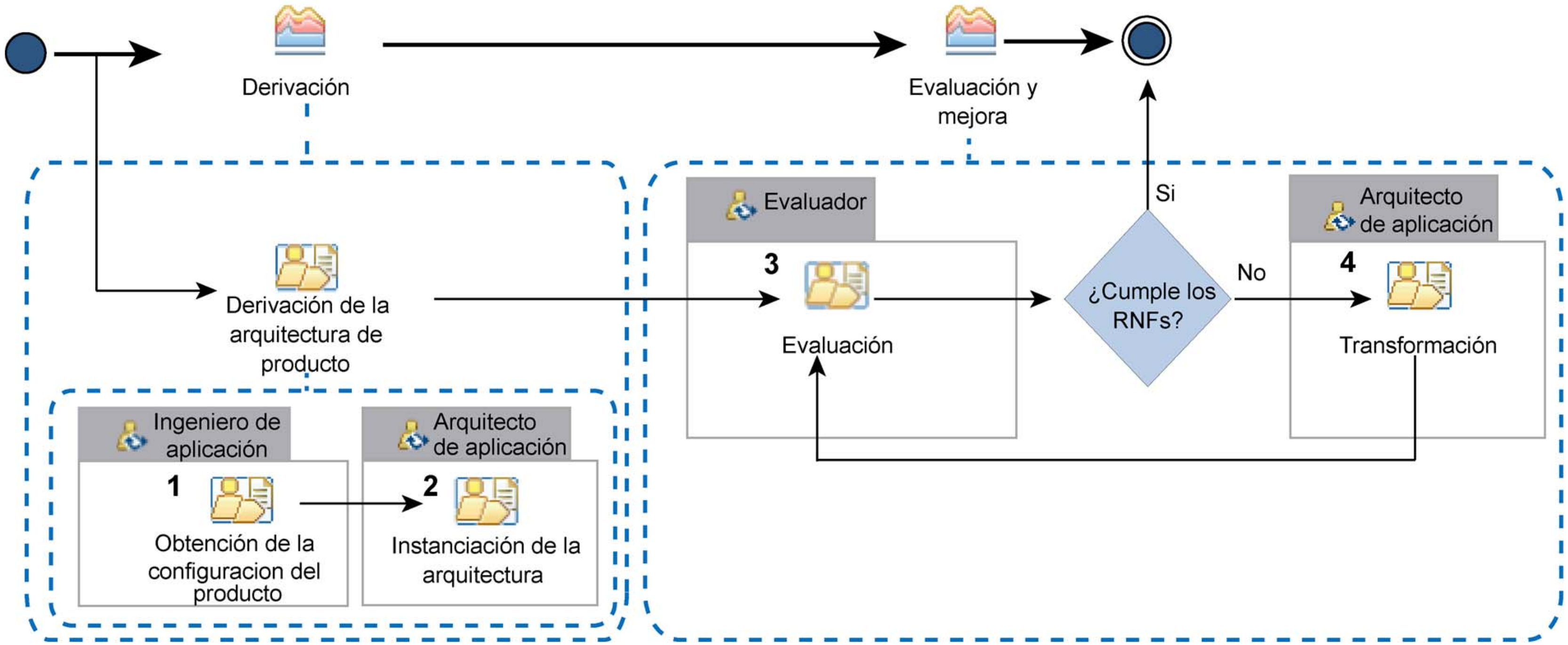
- The multimodel can be used to define relationships among the elements on different viewpoints
- This information expresses the missing information that the separation of concerns may introduce
- These relationships are used during the different tasks that integrate the QuaDAI derivation process



Relationships Used during Configuration

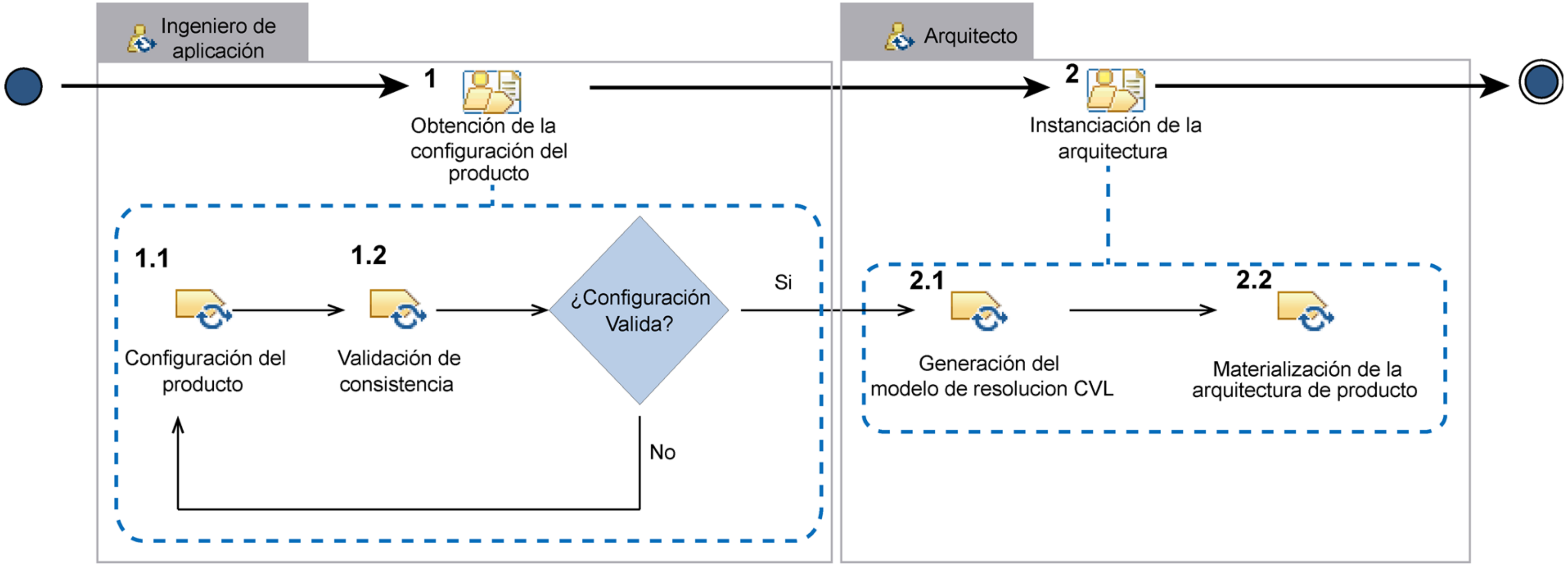
Relationships Used during Derivation

QuaDAI General Process

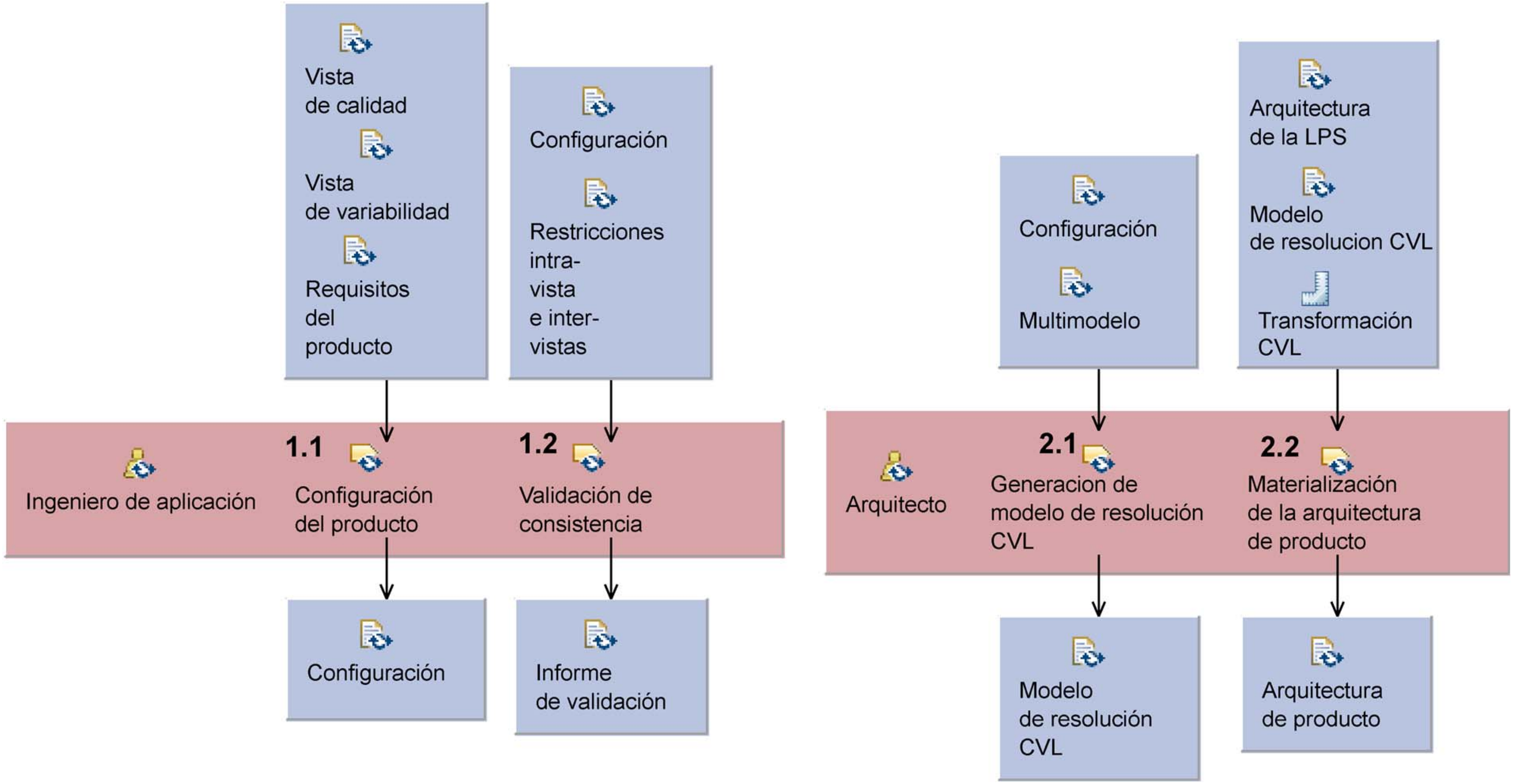


- In this case study we are going to focus only on the derivation phase.

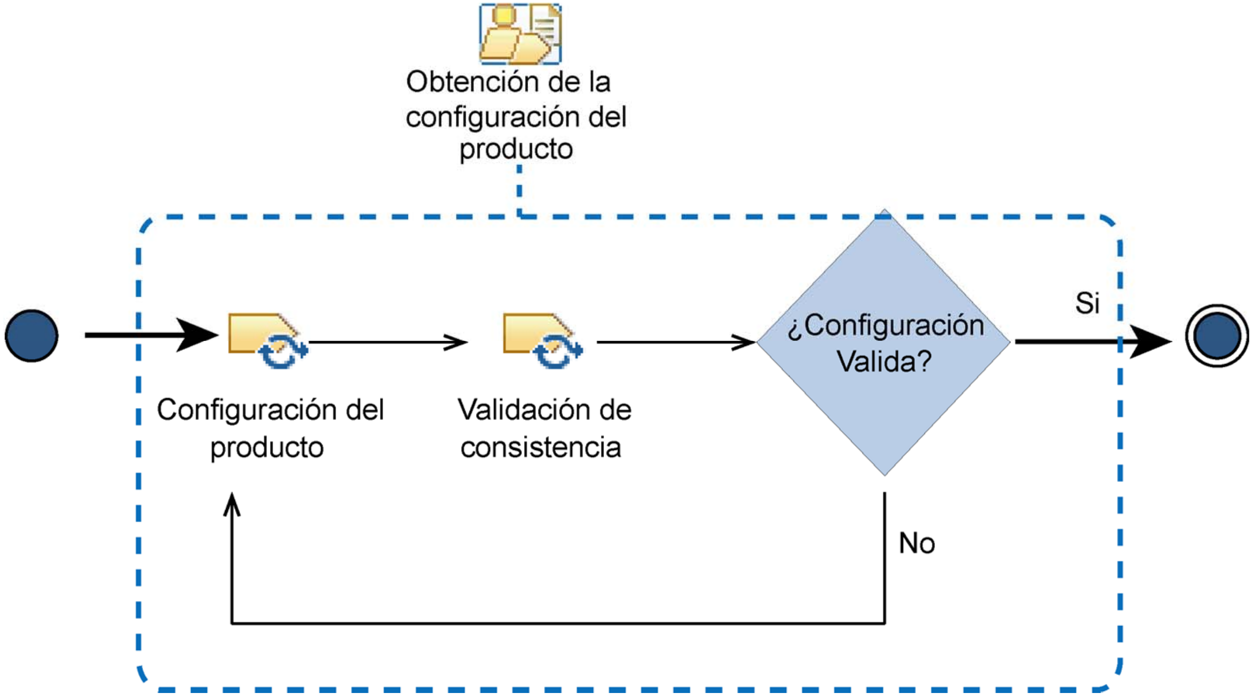
QuaDAI Derivation Process


Ingeniero de aplicación
Arquitecto
1
Obtención de la configuración del producto
2
Instanciación de la arquitectura
1.1
Configuración del producto
1.2
Validación de consistencia
¿Configuración Valida?
Si
No
2.1
Generación del modelo de resolucion CVL
2.2
Materialización de la arquitectura de producto

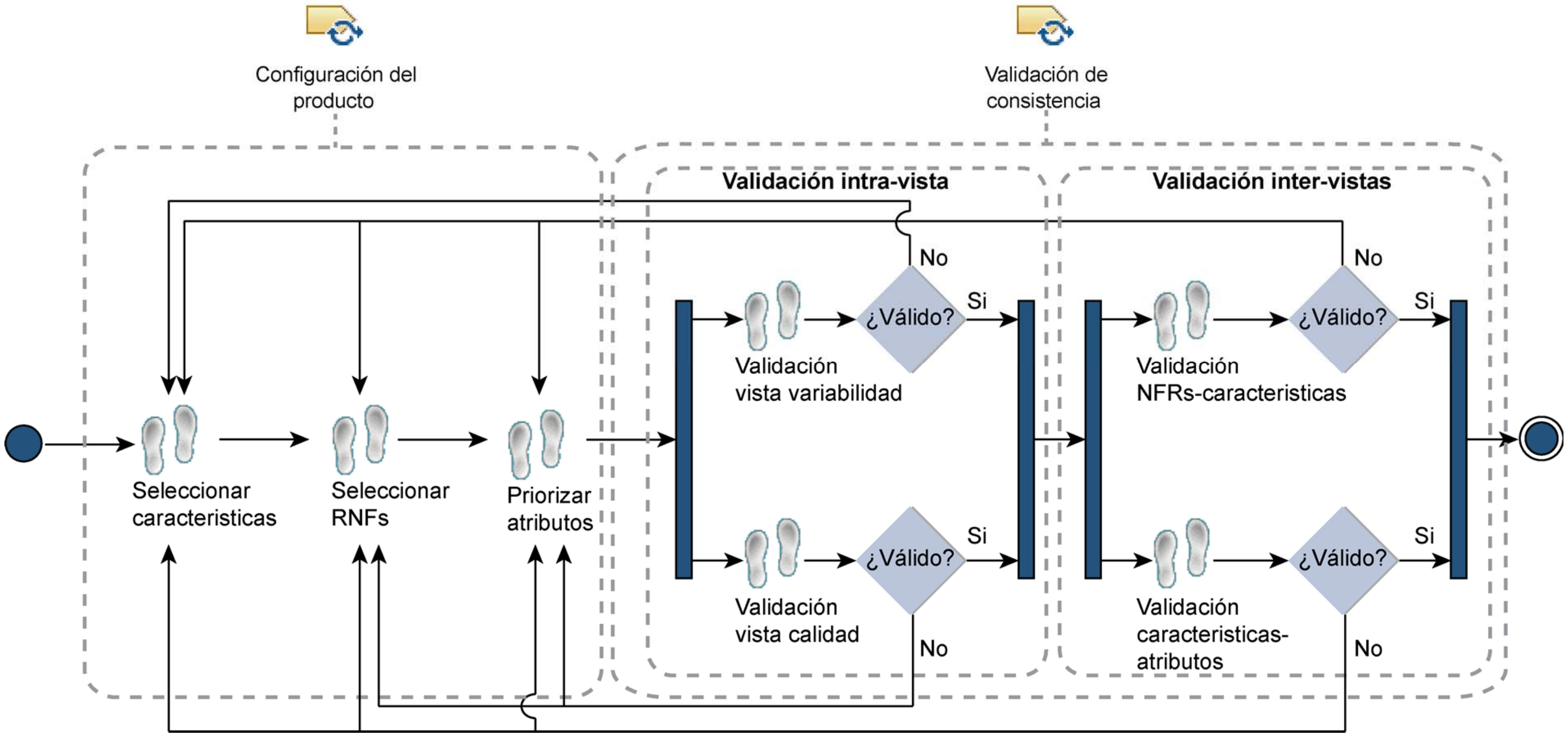
Detailed Activities



Configuration

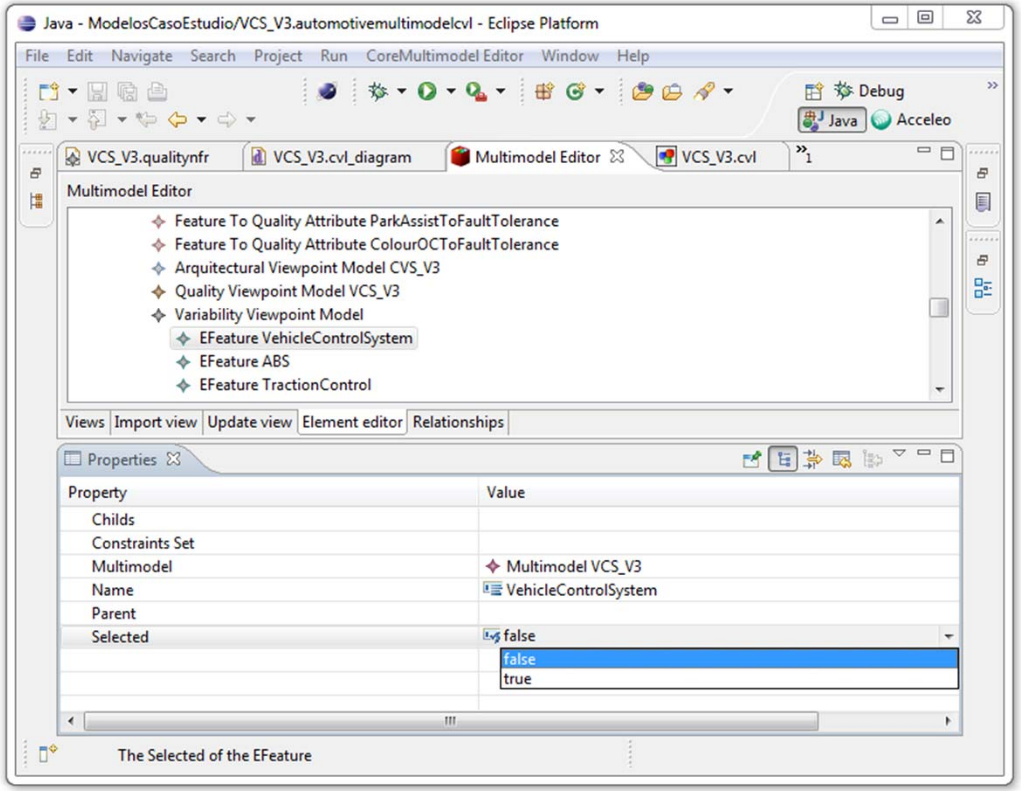

Obtención de la configuración del producto
Configuración del producto
Validación de consistencia
¿Configuración Valida?
Si
No

Configuration and consistency validation


Configuración del producto
Validación de consistencia
Validación intra-vista
Validación inter-vistas
Seleccionar caracteristicas
Seleccionar RNFs
Priorizar atributos
Validación vista variabilidad
Validación vista calidad
Validación NFRs-caracteristicas
Validación caracteristicas-atributos
¿Válido?
¿Válido?
¿Válido?
¿Válido?
Si
Si
Si
Si
No
No
No
No

Features Selection

- We have to select the features want the product to have by checking them in the tool.
  - We should select the root feature.
  - We should select a feature which as a child feature we are going to select



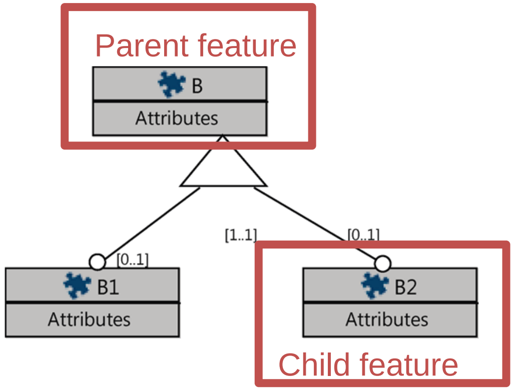
Java - ModelosCasoEstudio/VCS_V3.automotivemultimodelcvl - Eclipse Platform
File Edit Navigate Search Project Run CoreMultimodel Editor Window Help
Debug
Java
Acceleo
VCS_V3.qualitynfr
VCS_V3.cvl_diagram
Multimodel Editor
VCS_V3.cvl
Views
Import view
Update view
Element editor
Relationships
Properties

Multimodel Editor

- Feature To Quality Attribute ParkAssistToFaultTolerance
- Feature To Quality Attribute ColourOCToFaultTolerance
- Arquitectural Viewpoint Model CVS_V3
- Quality Viewpoint Model VCS_V3
- Variability Viewpoint Model
  - EFeature VehicleControlSystem
  - EFeature ABS
  - EFeature TractionControl

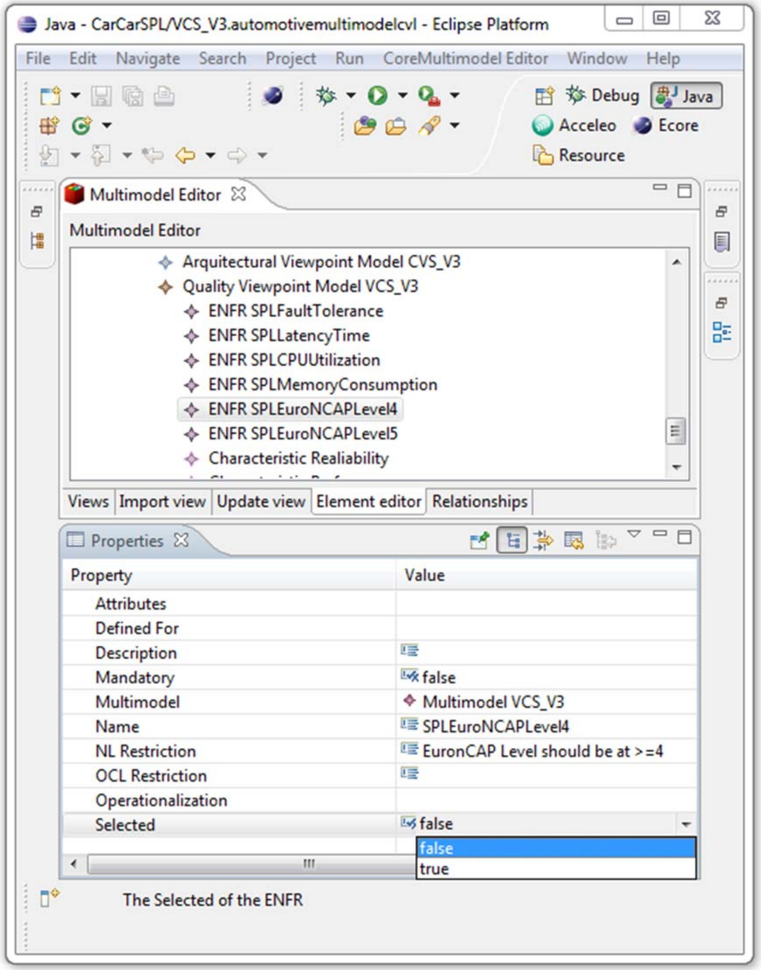
Property	Value
Childs	
Constraints Set	
Multimodel	Multimodel VCS_V3
Name	VehicleControlSystem
Parent	
Selected	false
	false true

The Selected of the EFeature



NFRs Selection

- Select the SPL and the product specific NFRs the product has to fulfill.
- If a product specific NFRs restricts a SPL's NFR, we have to select both
  - Due to the fact that the relationships with features are defined by using the SPL's NFR not with the product-specific NFRs.



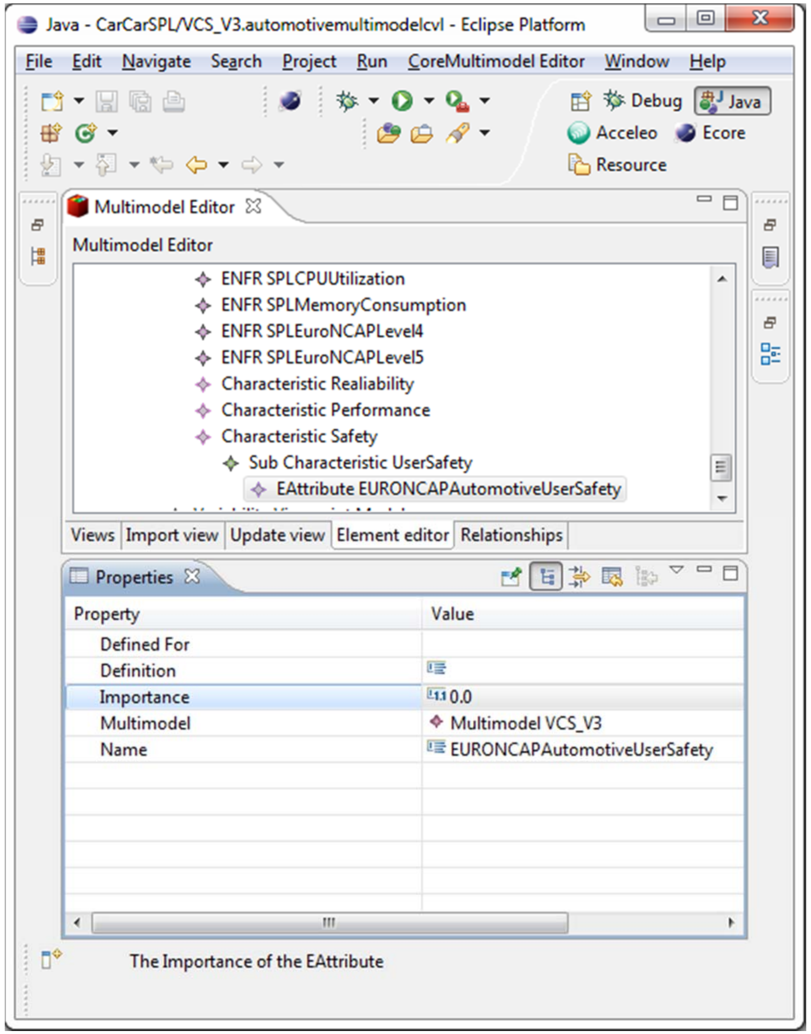
- Arquitectural Viewpoint Model CVS_V3
- Quality Viewpoint Model VCS_V3
  - ENFR SPLFaultTolerance
  - ENFR SPLLatencyTime
  - ENFR SPLCPUUtilization
  - ENFR SPLMemoryConsumption
  - ENFR SPLEuroNCAPLevel4
  - ENFR SPLEuroNCAPLevel5
  - Characteristic Realiability

Property	Value
Attributes	
Defined For	
Description	
Mandatory	false
Multimodel	Multimodel VCS_V3
Name	SPLEuroNCAPLevel4
NL Restriction	EuronCAP Level should be at >=4
OCL Restriction	
Operationalization	
Selected	false false true

The Selected of the ENFR

Prioritize Quality Attributes

- Priority the quality attributes, by introducing a value ranging from 0 to 1.
  - Expresses the fraction of 1 that you assign to each attribute (1 for critical 0 for trivial).
  - If a quality attribute is not important (or less important) keep it at 0 level.
  - Leave some degrees of freedom
    I. For quality attributes that are impacted negatively by other prioritized quality attributes.
    II. For quality attributes that, have certain importance, but are not constraints or requirements on the product.



Multimodel Editor

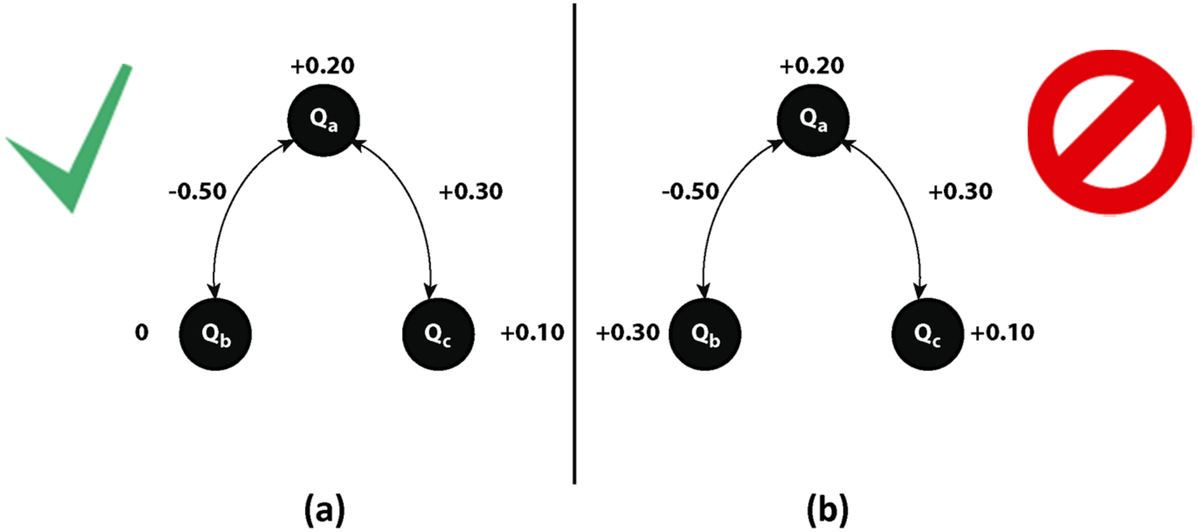
- ENFR SPLCPUUtilization
- ENFR SPLMemoryConsumption
- ENFR SPLEuroNCAPLevel4
- ENFR SPLEuroNCAPLevel5
- Characteristic Realiability
- Characteristic Performance
- Characteristic Safety
  - Sub Characteristic UserSafety
    - EAttribute EURONCAPAutomotiveUserSafety

Property	Value
Defined For	
Definition	
Importance	0.0
Multimodel	Multimodel VCS_V3
Name	EURONCAPAutomotiveUserSafety

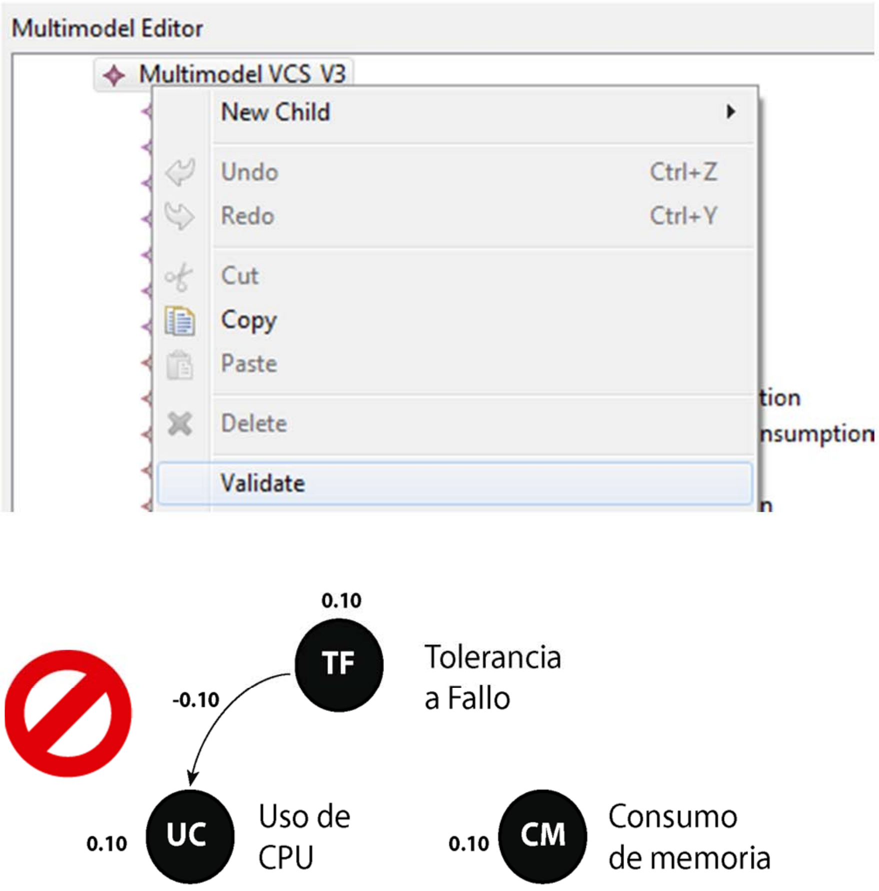
The Importance of the EAttribute

Quality Viewpoint Consistency

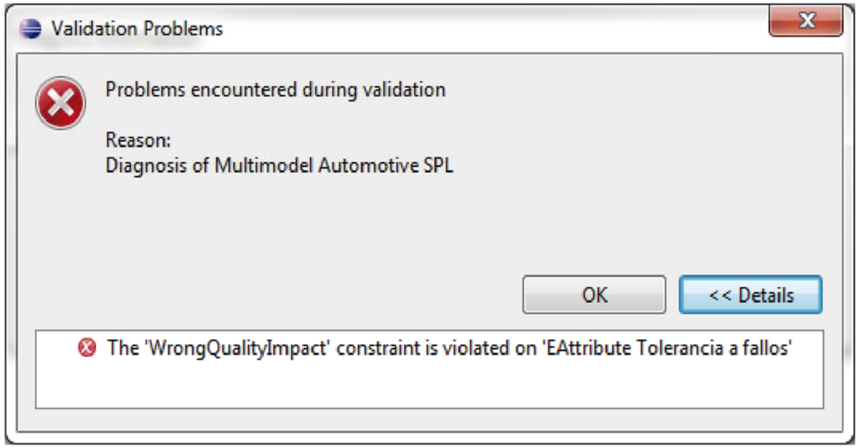
- For validating the consistency at the quality viewpoint we analyze that the prioritized quality attributes (QA) do not have negative impact relationships among them:
  - The (a) configuration there are no prioritized QAs that have negative impacts among them (Qb has no priority)
  - The (b) configuration includes a pair of QAs which impact negatively one on the other (Qa and Qb)



Quality Viewpoint Consistency Validation



(a)



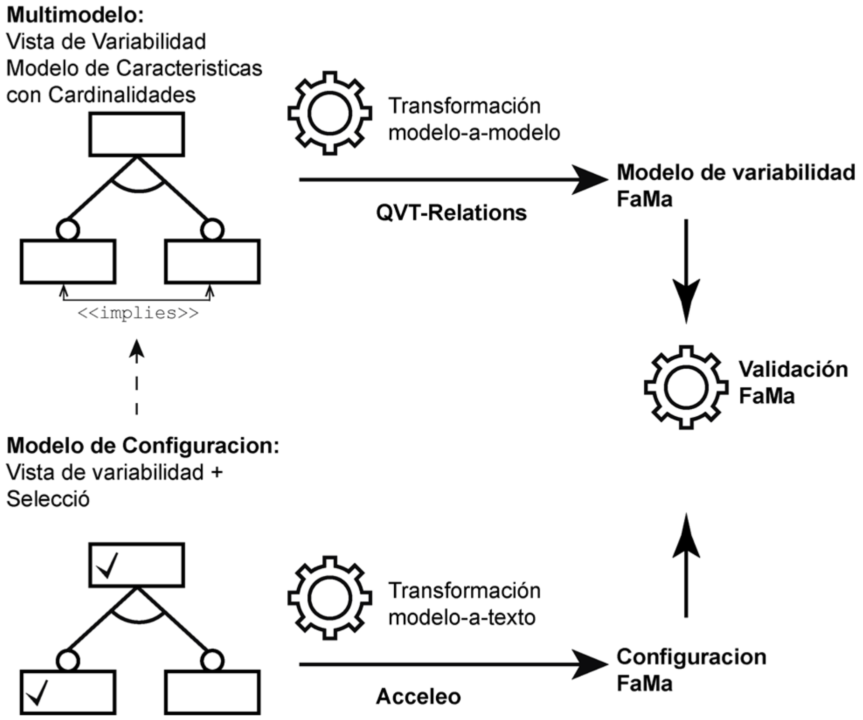
Problems encountered during validation

Reason:
Diagnosis of Multimodel Automotive SPL

The 'WrongQualityImpact' constraint is violated on 'EAttribute Tolerancia a fallos'

(b)

Variability Consistency Validation

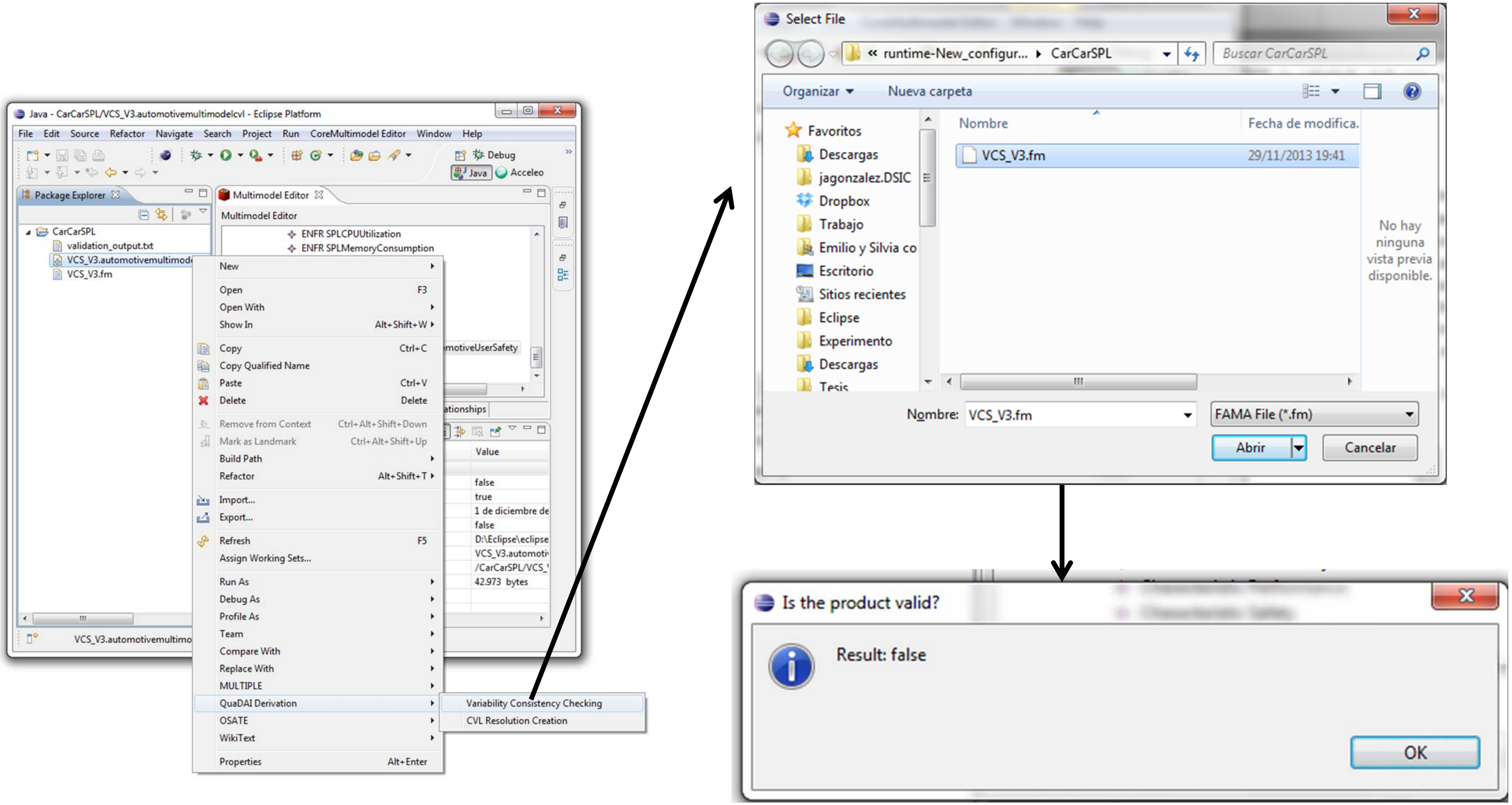


- The variability consistency validation checks the conformance of the selected set of features with the constraints and restrictions defined in the feature model
  - We translate the Feature Model to the FaMa Tool[1] representation
  - We inject the selected features to the FaMa validator and obtain whether the set of features is a valid configuration or not

[1] FaMa Framework© ISA research group

http://www.isa.us.es/fama/

Variability Consistency Checking

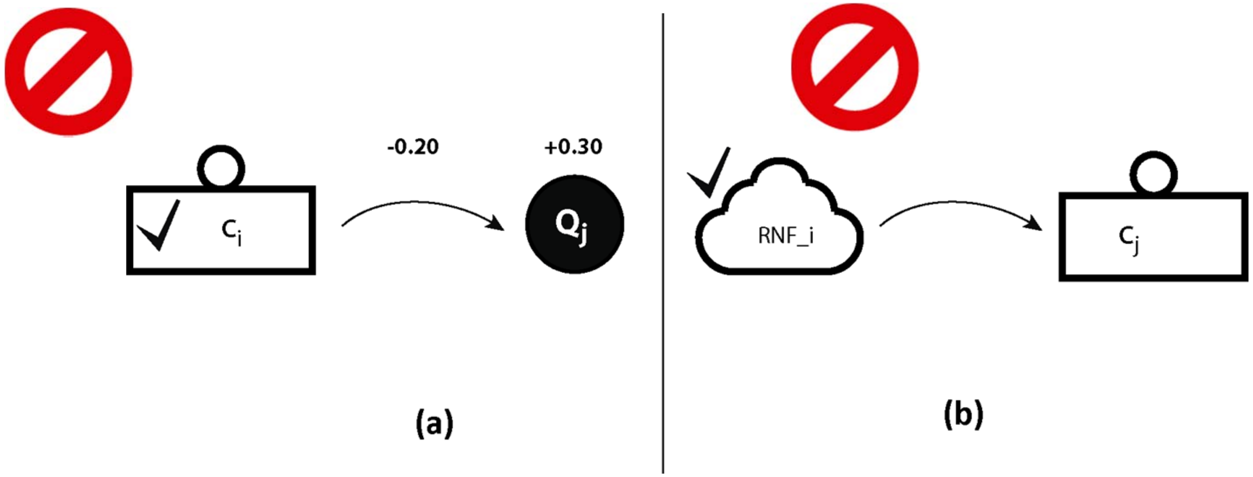


Is the product valid?

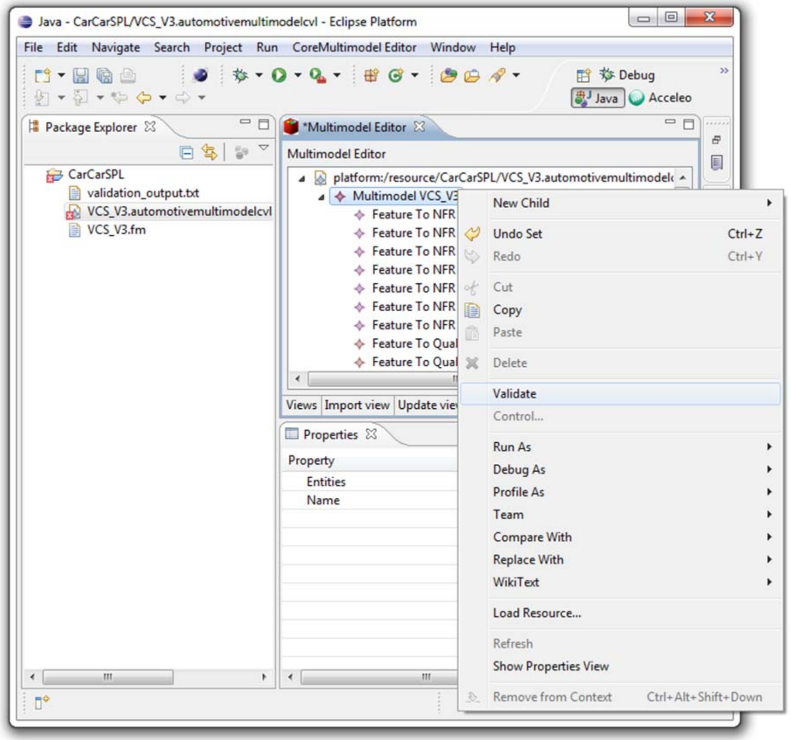
Result: false

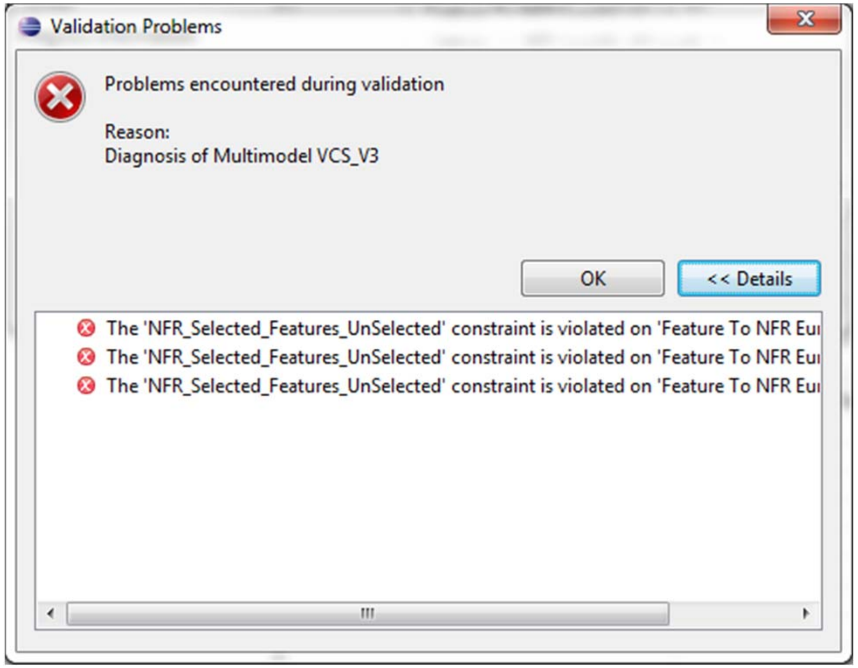
Inter-Viewpoint Consistency Checking

- The consistency among views should be checked to assure that the selected features, NFRs and priorities of quality attributes meet the constraints we have defined in the multimodel through the multimodel relationships
- We check two main facts:
  - That there is no feature selected witch impacts negatively to a prioritized quality attribute
  - That all the features that realize the selected NFRs had been selected.



Inter-Viewpoint Consistency Checking





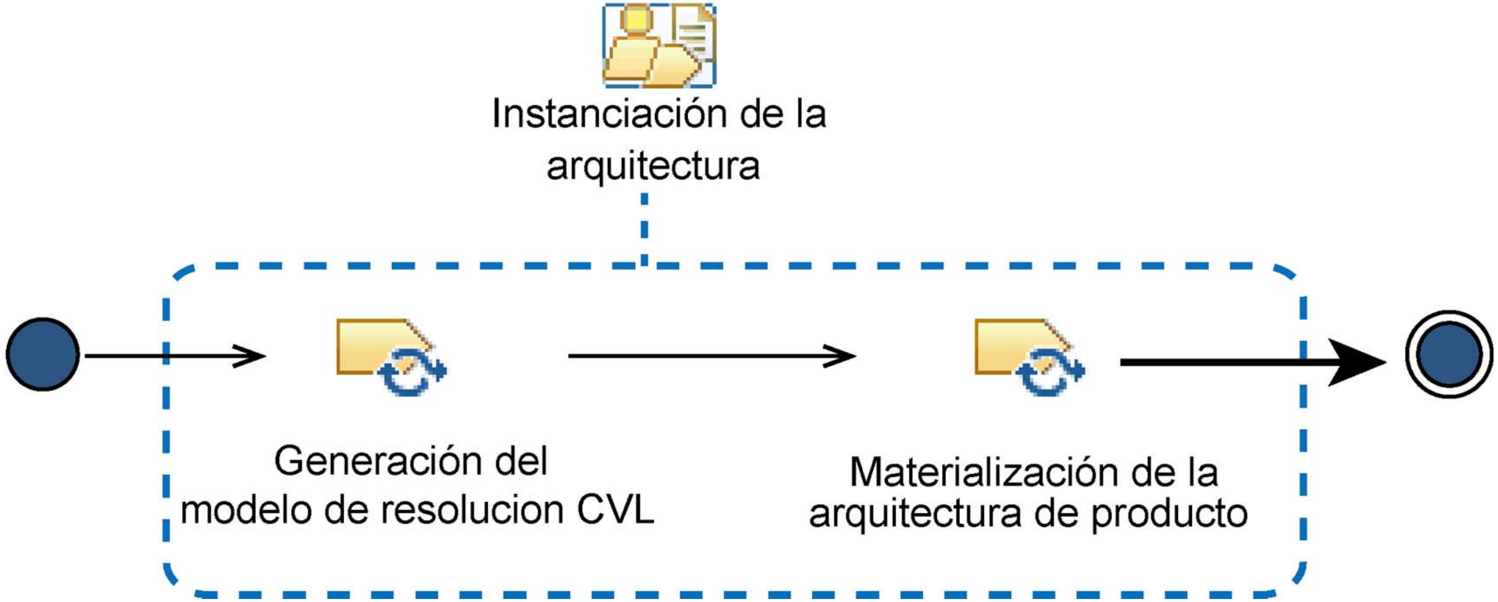
Problems encountered during validation

Reason:
Diagnosis of Multimodel VCS_V3

The 'NFR_Selected_Features_UnSelected' constraint is violated on 'Feature To NFR Eu

The 'NFR_Selected_Features_UnSelected' constraint is violated on 'Feature To NFR Eu

The 'NFR_Selected_Features_UnSelected' constraint is violated on 'Feature To NFR Eu



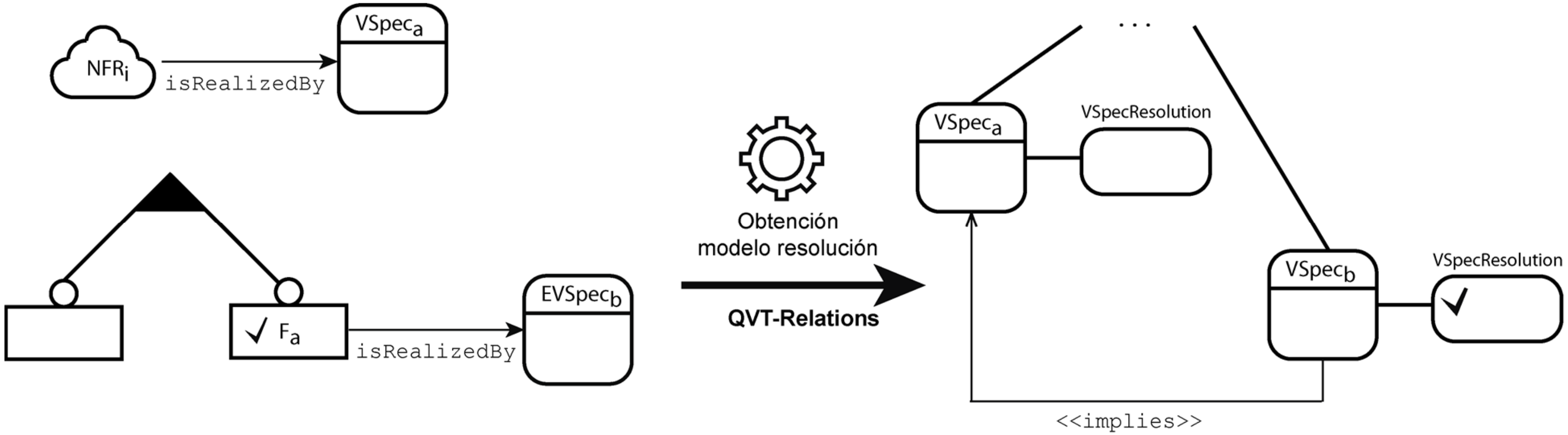
Instanciación de la arquitectura

Generación del modelo de resolucion CVL

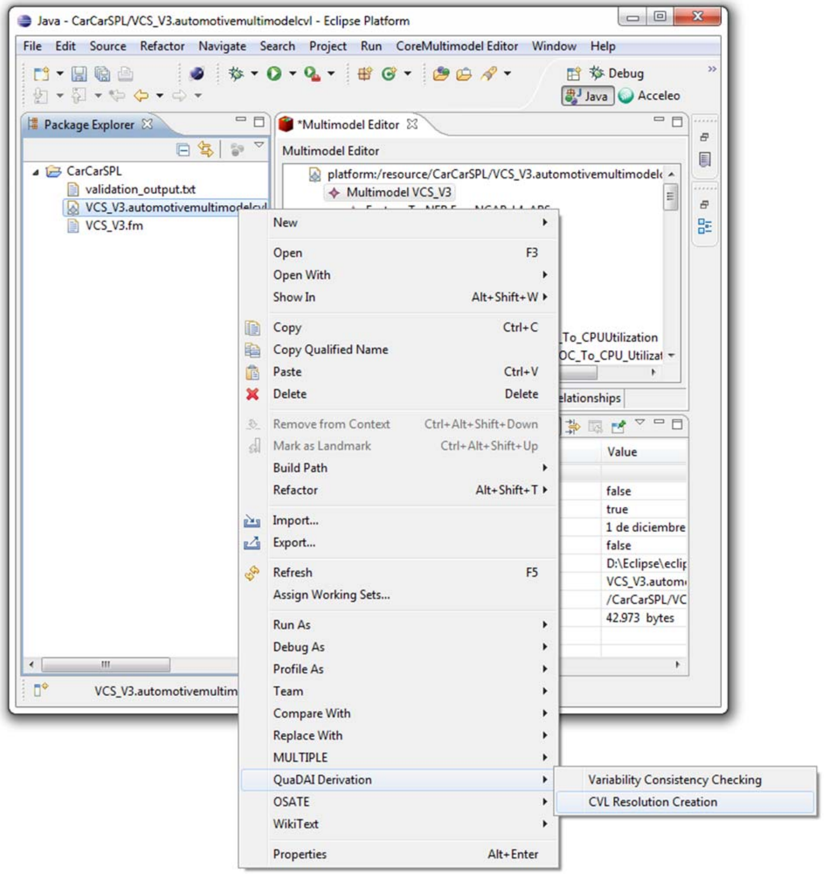
Materialización de la arquitectura de producto

CVL Resolution Model Generation

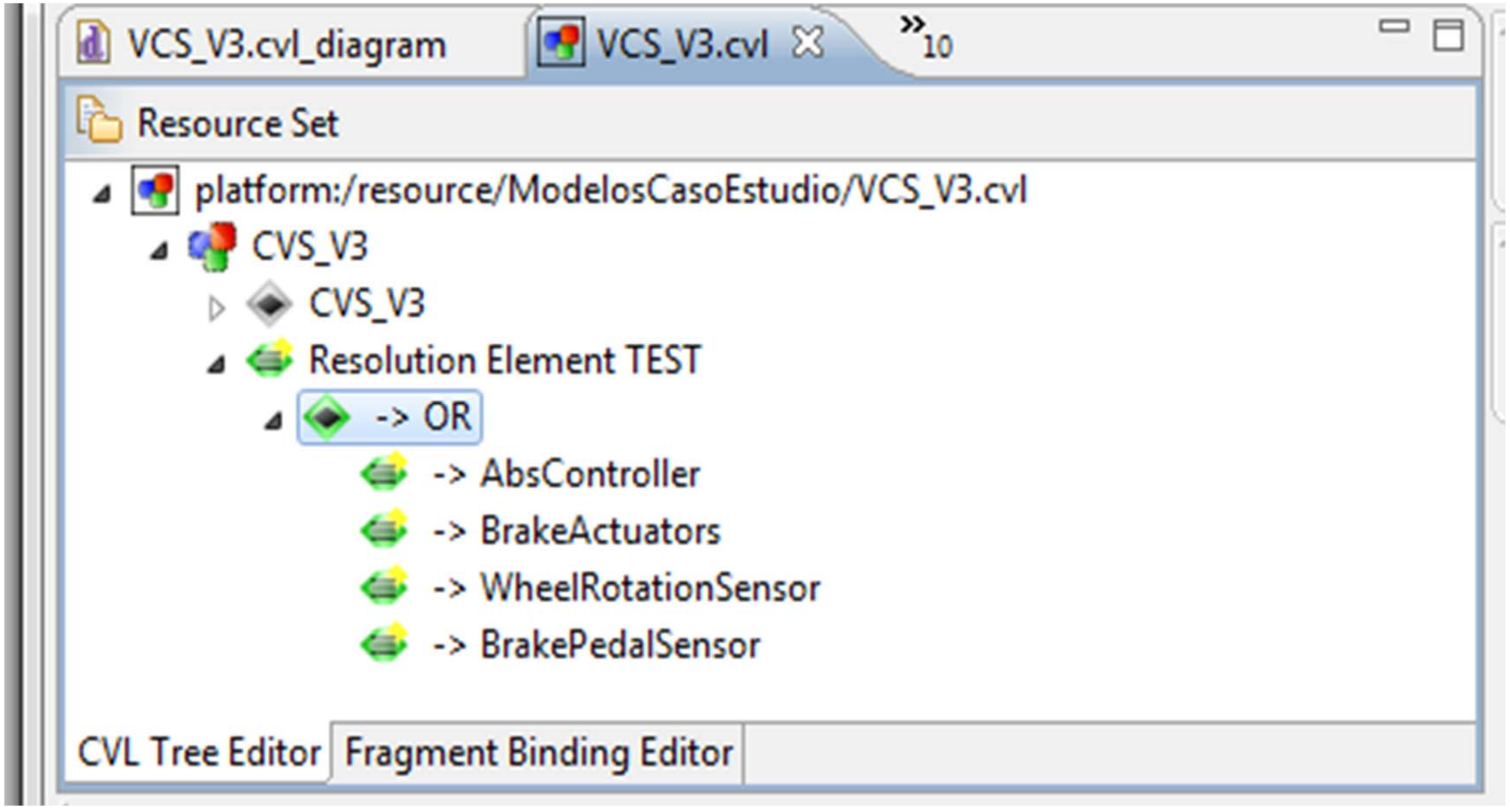
- The relationships among architectural variation points, features, non-functional requirements and quality attributes, are used now to derive the CVL resolution model that will allow us to obtain the first version of the architecture.



CVL Resolution Model Generation

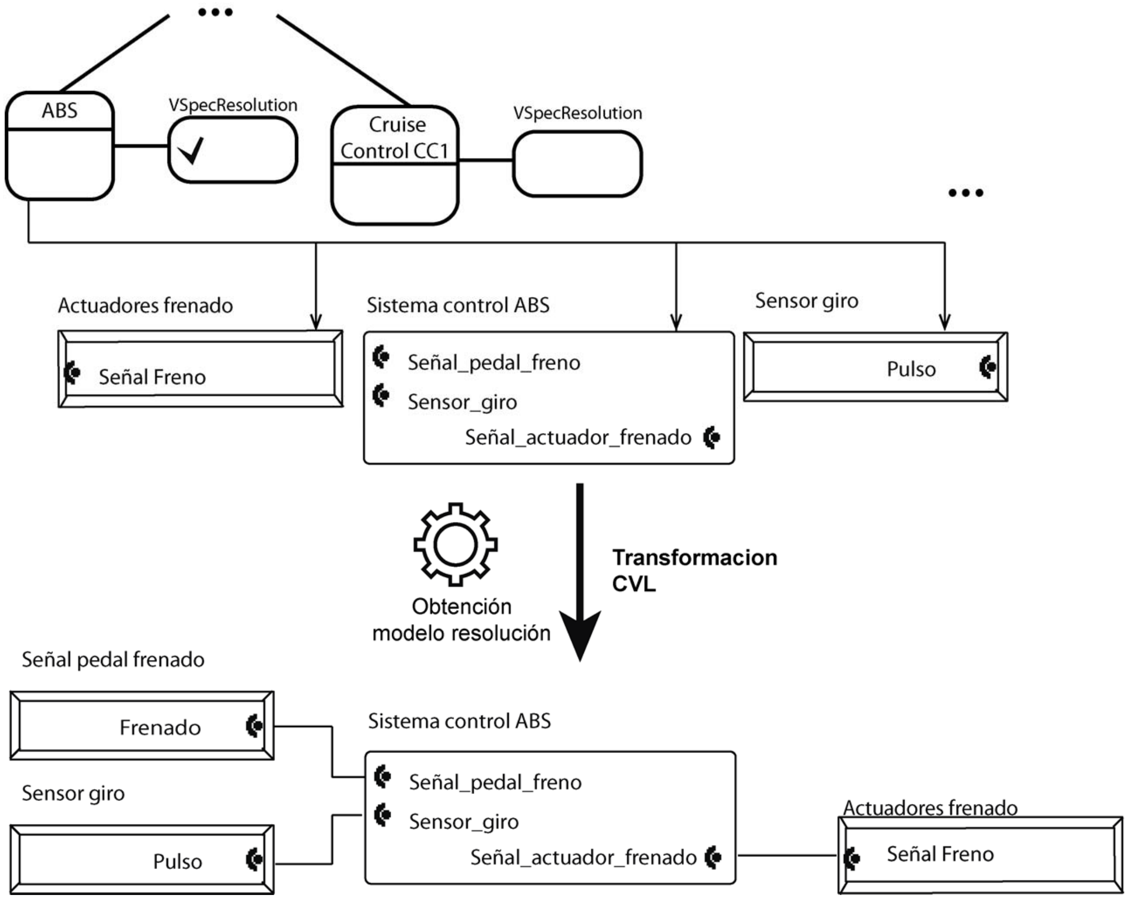


CVL Resolution Model Obtained

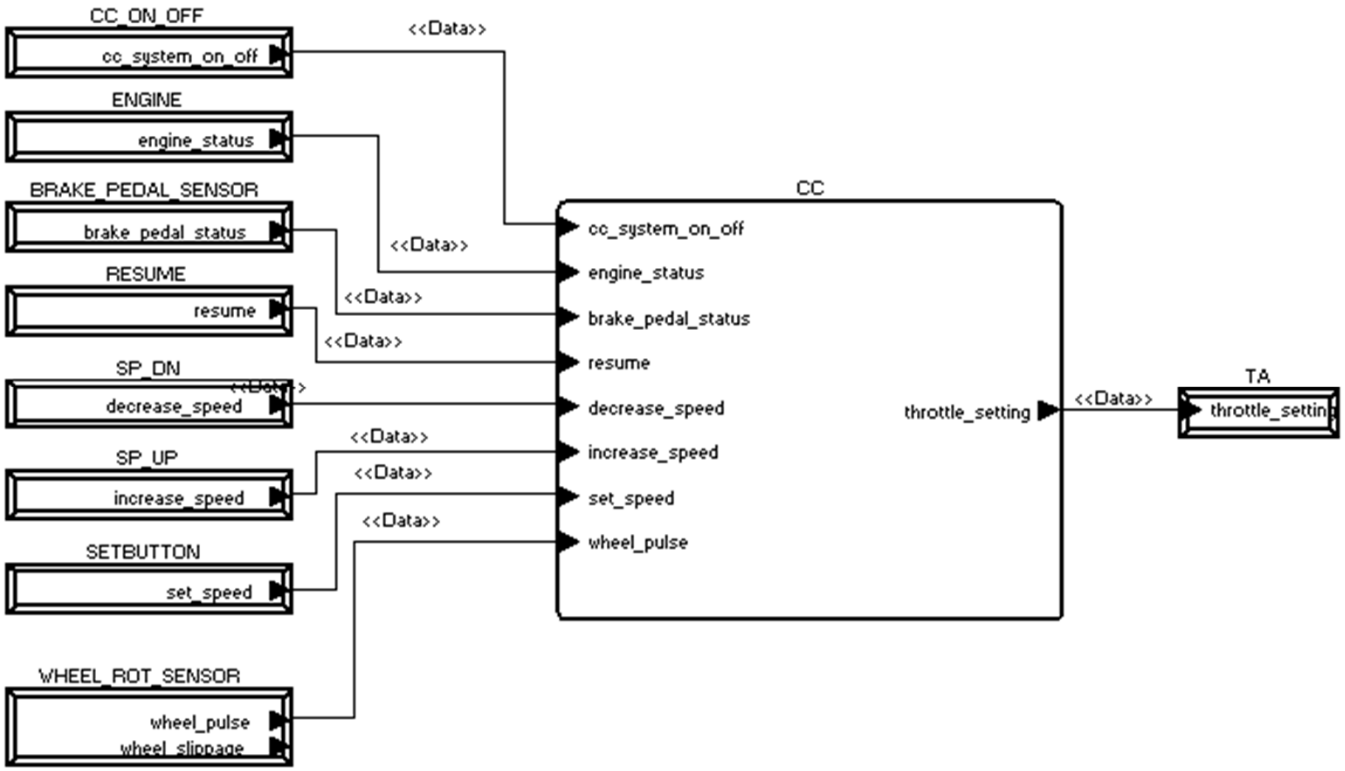


```
platform:/resource/ModelosCasoEstudio/VCS_V3.cvl
  CVS_V3
    CVS_V3
    Resolution Element TEST
      -> OR
        -> AbsController
        -> BrakeActuators
        -> WheelRotationSensor
        -> BrakePedalSensor
```

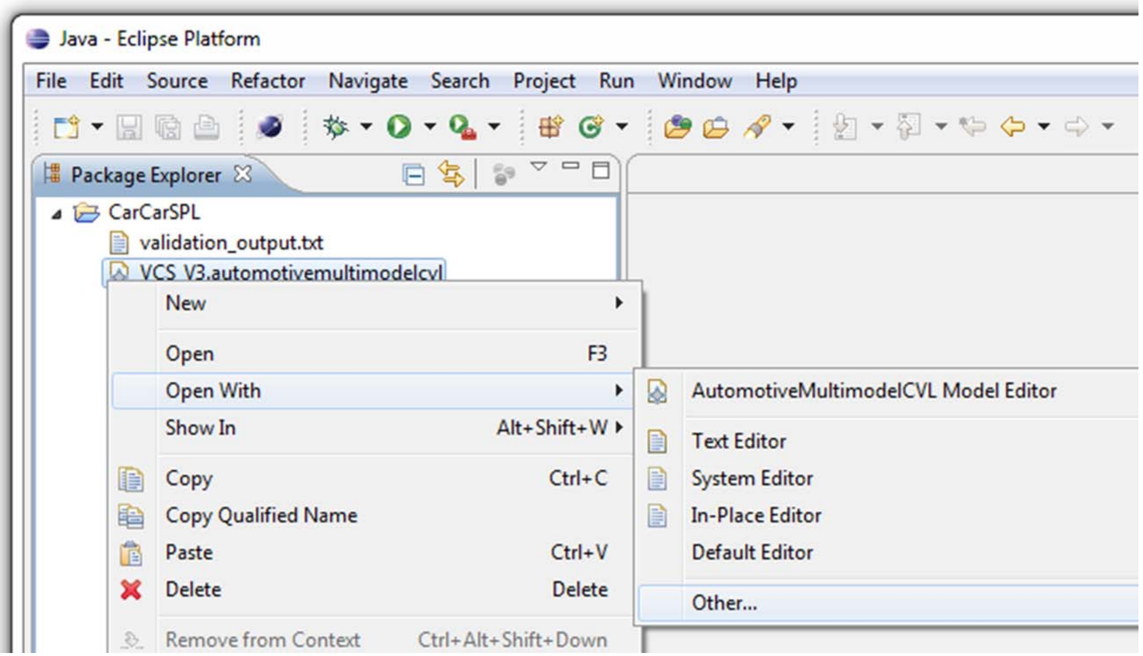
Architecture Materialization

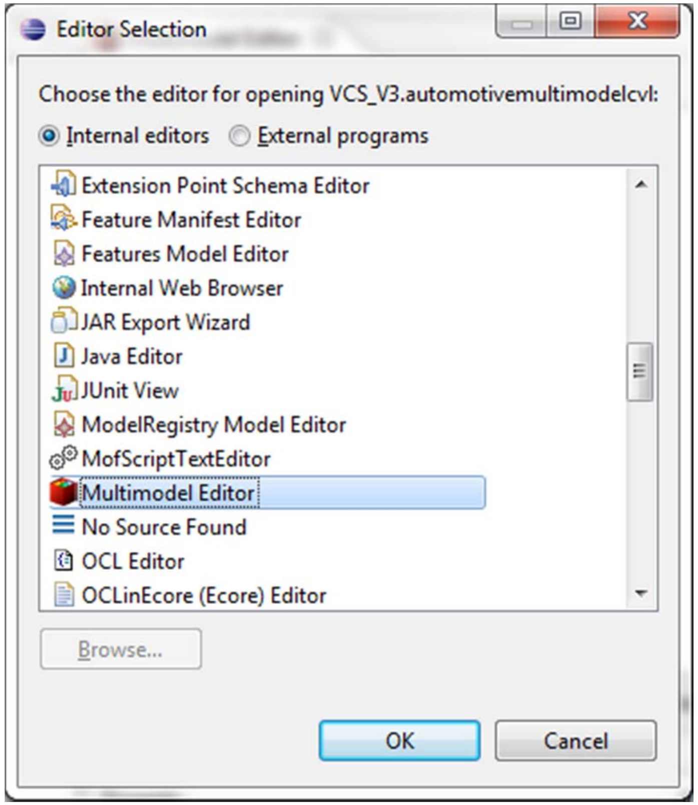


Example of the Materialized Architecture



Practical Info: The multimodel Editor

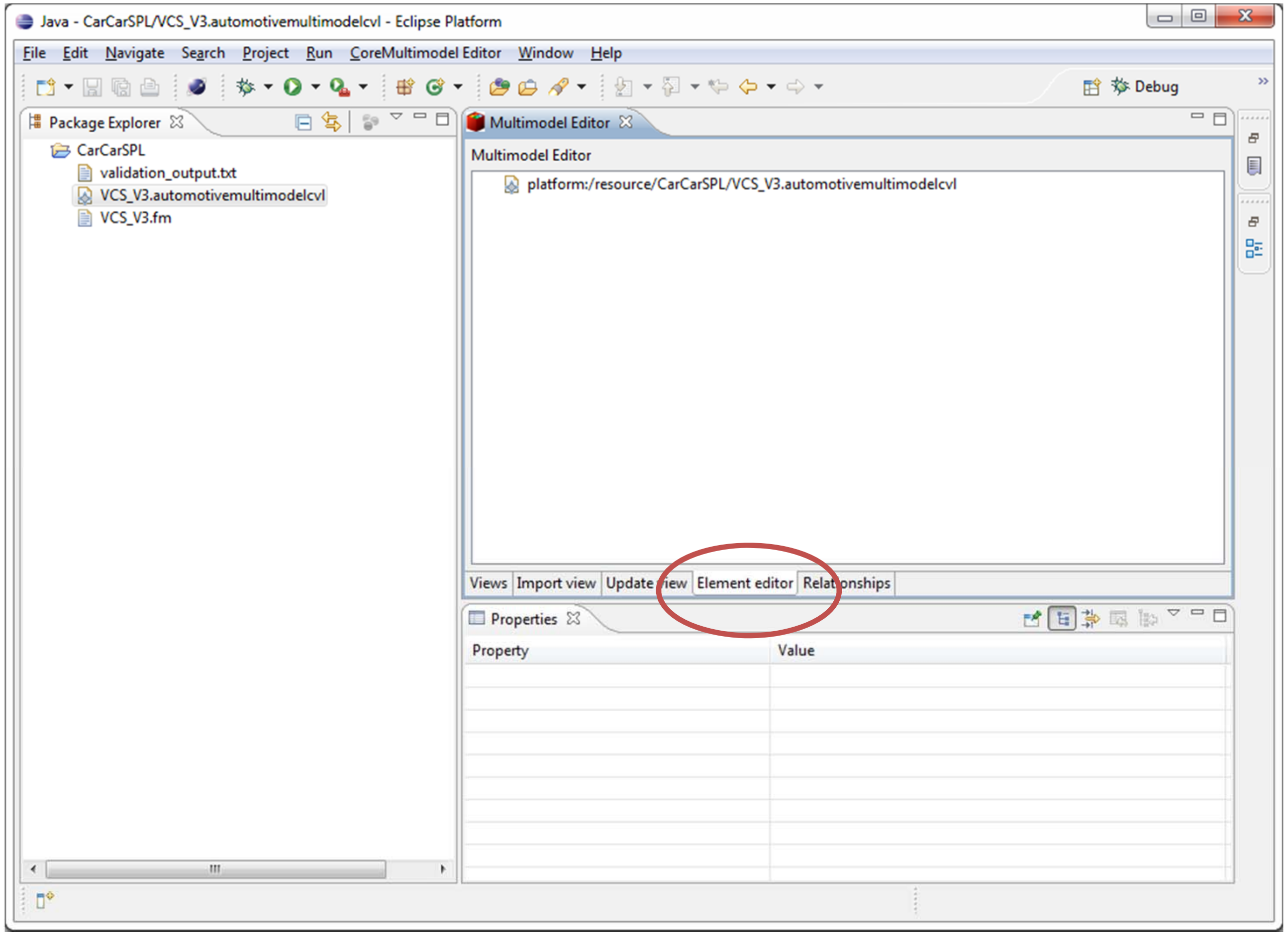




Choose the editor for opening VCS_V3.automotivemultimodelcvl:

● Internal editors ○ External programs

The element view


Java - CarCarSPL/VCS_V3.automotivemultimodelcvl - Eclipse Platform
File Edit Navigate Search Project Run CoreMultimodel Editor Window Help
Debug
Package Explorer
CarCarSPL
validation_output.txt
VCS_V3.automotivemultimodelcvl
VCS_V3.fm
Multimodel Editor
Multimodel Editor
platform:/resource/CarCarSPL/VCS_V3.automotivemultimodelcvl
Views
Import view
Update view
Element editor
Relationships
Properties
Property
Value