UNIVERSITY OF Southampton

deploy

An Atomicity Decomposition Technique for Event-B

Asieh Salehi

Michael Butler

(asf08r, mjb@ecs.soton.ac.uk)

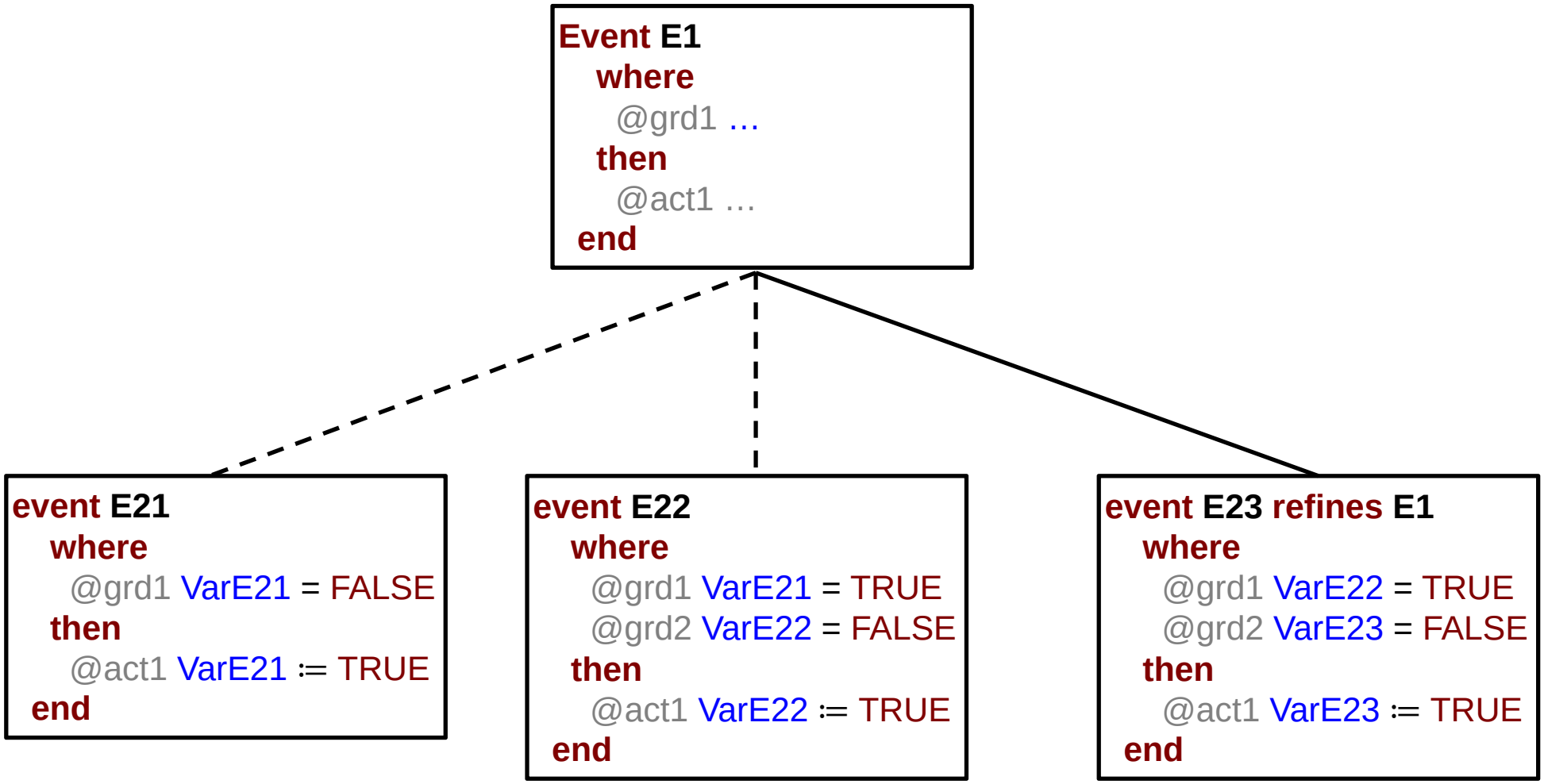
Outline

- Introduction and Motivation
- Atomicity Decomposition Diagram
- Translation Rules
- Case Study 1: Multi Media Protocol
- Case Study 2: Space Craft
- Tool Development
- Conclusion

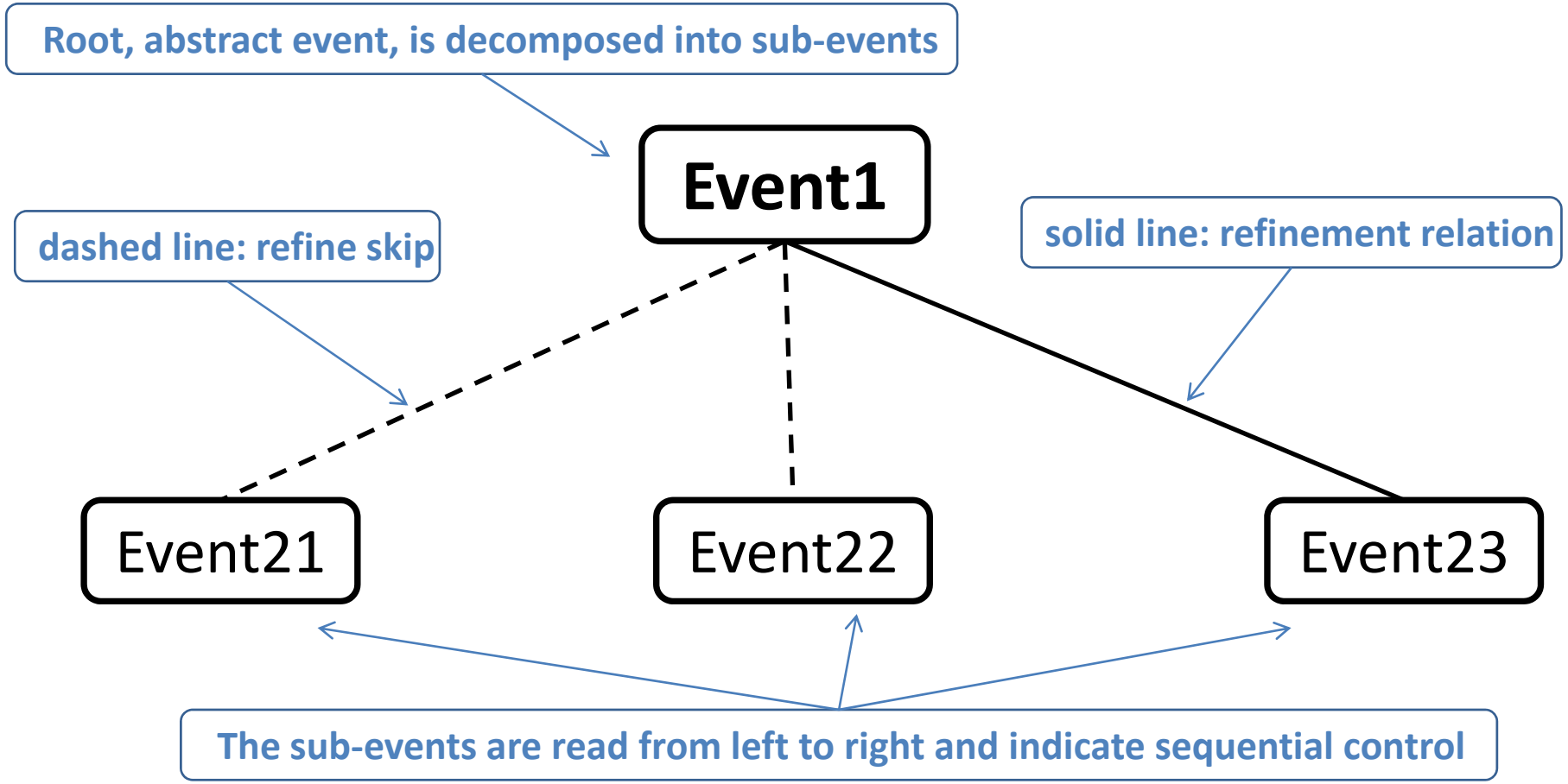
Introduction

- Problem: large and complex models and proofs
- Some solutions : Refinement, Generic instantiation, Decomposition
- Decomposition in Event-B :
  - Model Decomposition
  - **Atomicity Decomposition** : a technique augmented to Refinement in Event-B

Motivation

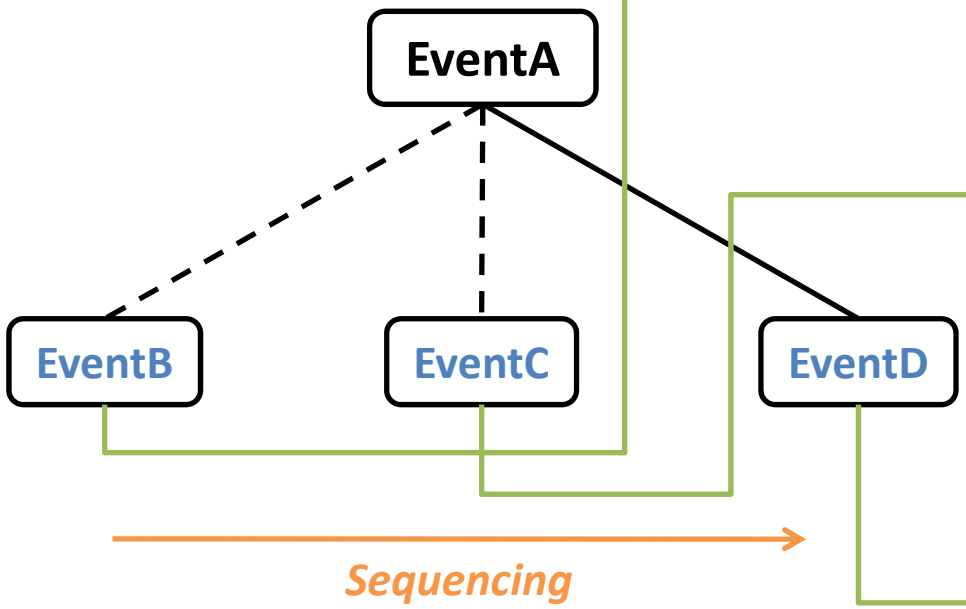

Event E1
where
@grd1 …
then
@act1 …
end
event E21
where
@grd1 VarE21 = FALSE
then
@act1 VarE21 ≔ TRUE
end
event E22
where
@grd1 VarE21 = TRUE
@grd2 VarE22 = FALSE
then
@act1 VarE22 ≔ TRUE
end
event E23 refines E1
where
@grd1 VarE22 = TRUE
@grd2 VarE23 = FALSE
then
@act1 VarE23 ≔ TRUE
end

Atomicity Decomposition Diagram*



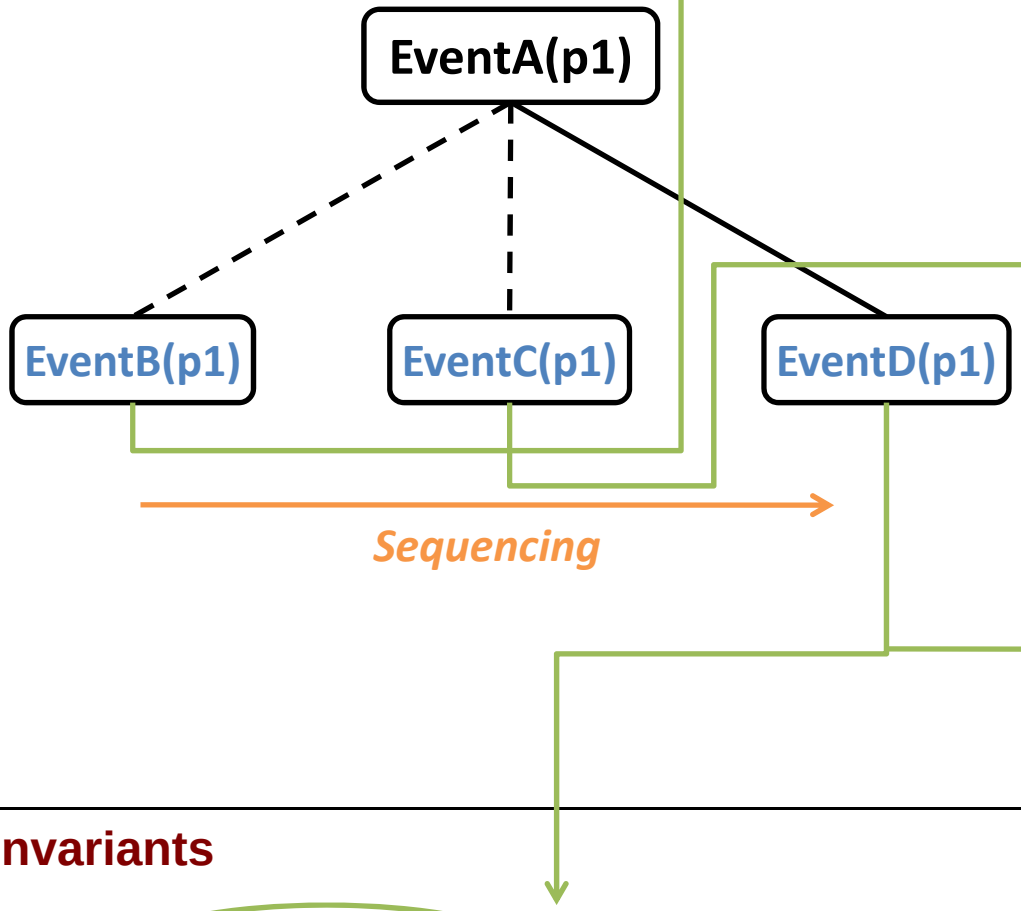
* Butler, M. (2009) Decomposition Structures for Event-B. In: Integrated Formal Methods iFM2009, Springer, LNCS 5423.

Translation Rules




```
event EventB
  where
    @grd1 EventB_Bool = FALSE
  then
    @act1 EventB_Bool ≔ TRUE
end

event EventC
  where
    @grd1 EventB_Bool = TRUE
    @grd2 EventC_Bool = FALSE
  then
    @act1 EventC_Bool ≔ TRUE
end

event EventD refines EventA
  where
    @grd1 EventC_Bool = TRUE
    @grd2 EventD_Bool = FALSE
  then
    @act1 EventD_Bool ≔ TRUE
end
```

Translation Rules



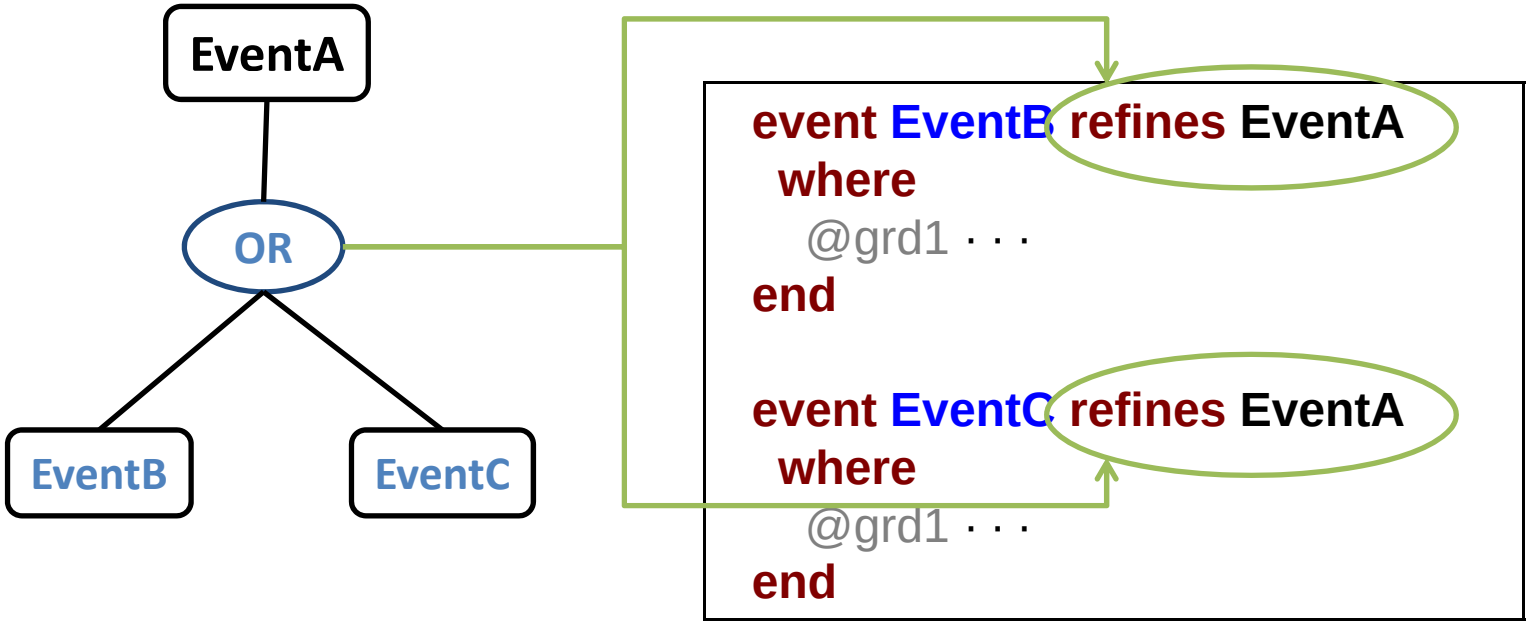

```
event EventB
  any p1
  where
    @grd1 p1 ∈ PARAMETERS \ EventB
  then
    @act1 EventB ≔ EventB ∪ {p1}
end

event EventC
  any p1
  where
    @grd1 p1 ∈ EventB \ EventC
  then
    @act1 EventC ≔ EventC ∪ {p1}
end

event EventD refines EventA
  any p1
  where
    @grd1 p1 ∈ EventC \ EventD
  then
    @act1 EventD ≔ EventD ∪ {p1}
end
```

Invariants

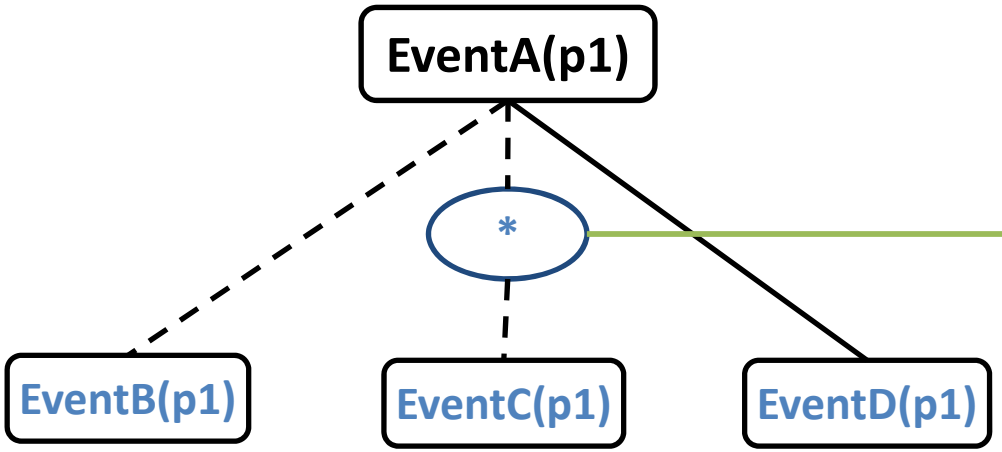
…

@inv3 EventD ⊆ EventA

Case Splitting

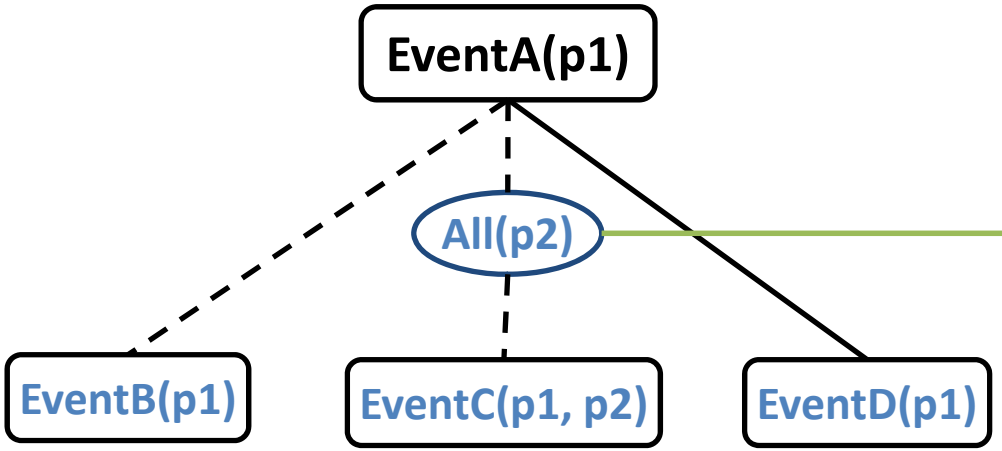


Loop



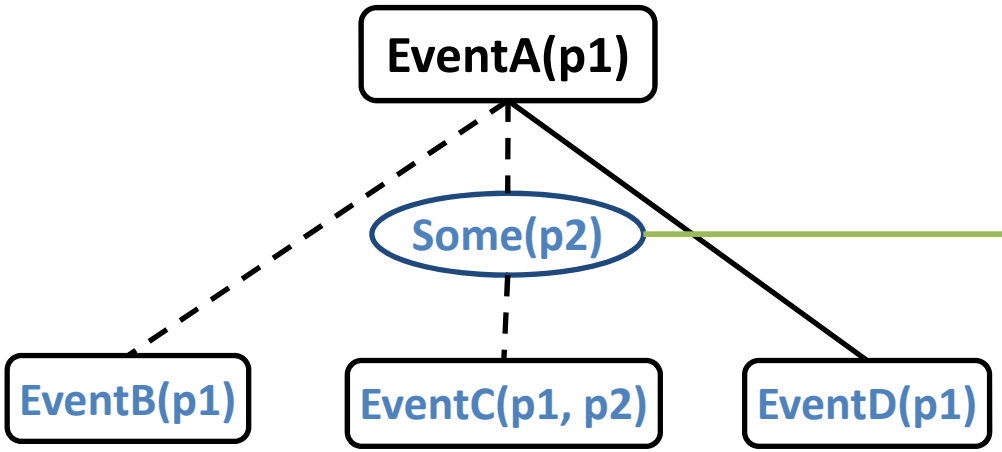

```
event EventB
  any p1
  where
    @grd1 p1 ∈ PARAMETERS \ EventB
  then
    @act1 EventB ≔ EventB ∪ {p1}
end

event EventC
  any p1
  where
    @grd1 p1 ∈ EventB \ EventD
  then
    @act1 . . .
end

event EventD refines EventA
  any p1
  where
    @grd1 p1 ∈ EventB \ EventD
  then
    @act1 EventD ≔ EventD ∪ {p1}
end
```

Replicator, *All*



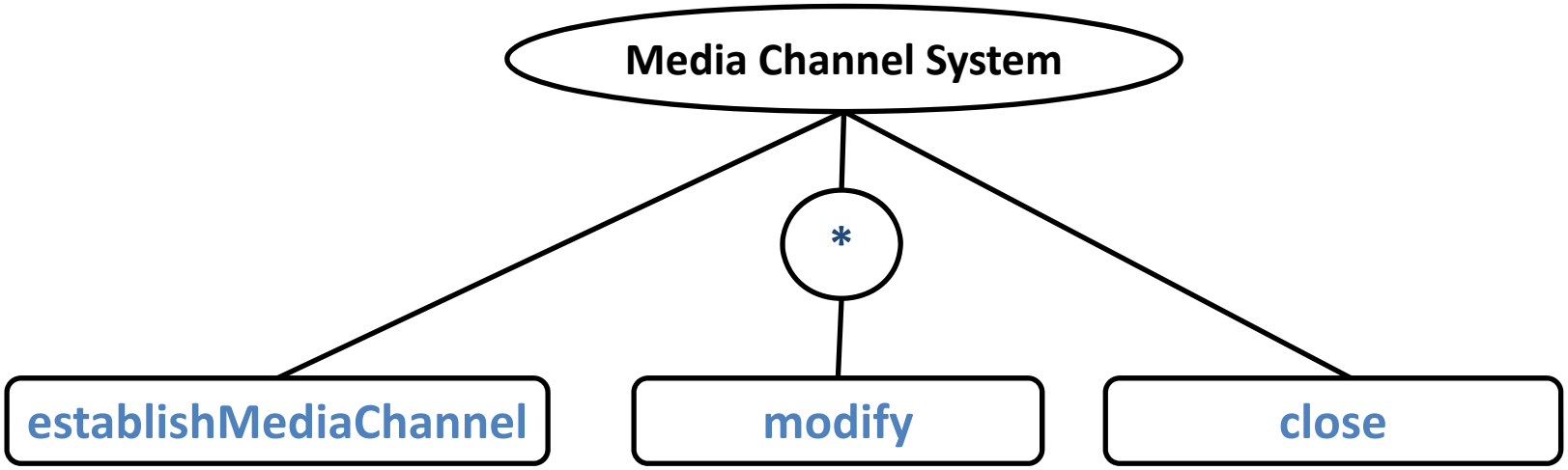

```
event EventB
  any p1
  where
    @grd1 p1 ∈ PARAMETERS \ EventB
  then
    @act1 EventB ≔ EventB ∪ {p1}
 end

event EventC
  any p1, P2
  where
    @grd1 p1 ∈ EventB
    @grd2 p1 ↦ p2 ∉ EventC
  then
    @act1 EventC ≔ EventC ∪ {p1 ↦ p2}
end

event EventD refines EventA
  any p1
  where
    @grd1 p1 ∉ EventD
    @grd2 ∀p2. p1 ↦ p2 ∈ EventC
 then
    @act1 EventD ≔ EventD ∪ {p1}
 end
```

Replicator, *Some*




```
event EventB
  any p1
  where
    @grd1 p1 ∈ PARAMETERS \ EventB
  then
    @act1 EventB ≔ EventB ∪ {p1}
end

event EventC
  any p1, P2
  where
    @grd1 p1 ∈ EventB
    @grd2 p1↦p2 ∉ EventC
  then
    @act1 EventC ≔ EventC ∪ {p1↦p2}
end

event EventD refines EventA
  any p1
  where
    @grd1 p1 ∉ EventD
    @grd2 ∃p2. p1↦p2 ∈ EventC
  then
    @act1 EventD ≔ EventD ∪ {p1}
end
```

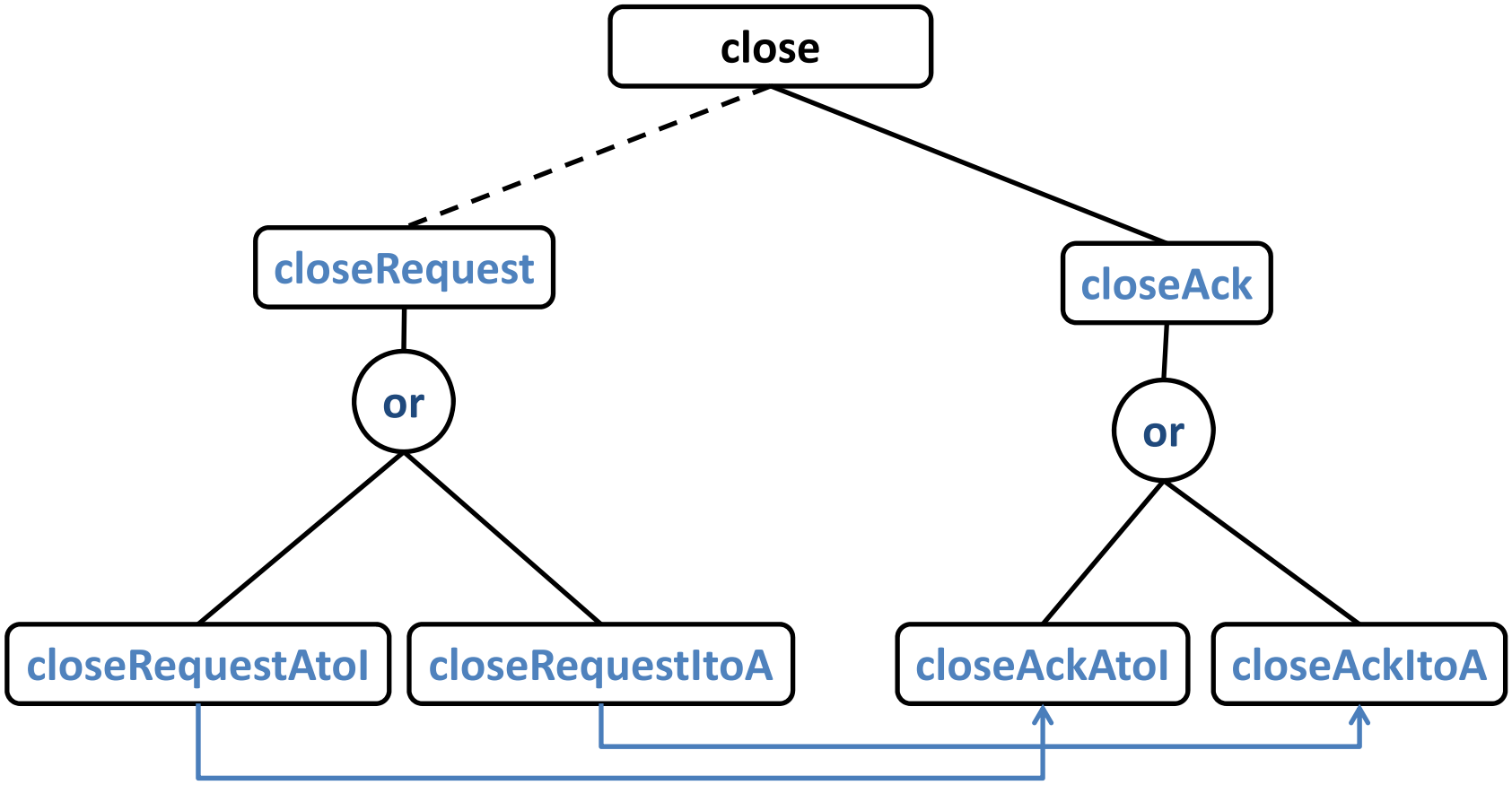
Multi Media Protocol



- 1st Refinement: Breaking the Atomicity of Establish Media Channel
- 2nd Refinement: Breaking the Atomicity of Modify Media Channel
- 3rd Refinement: Breaking the Atomicity of Close Media Channel

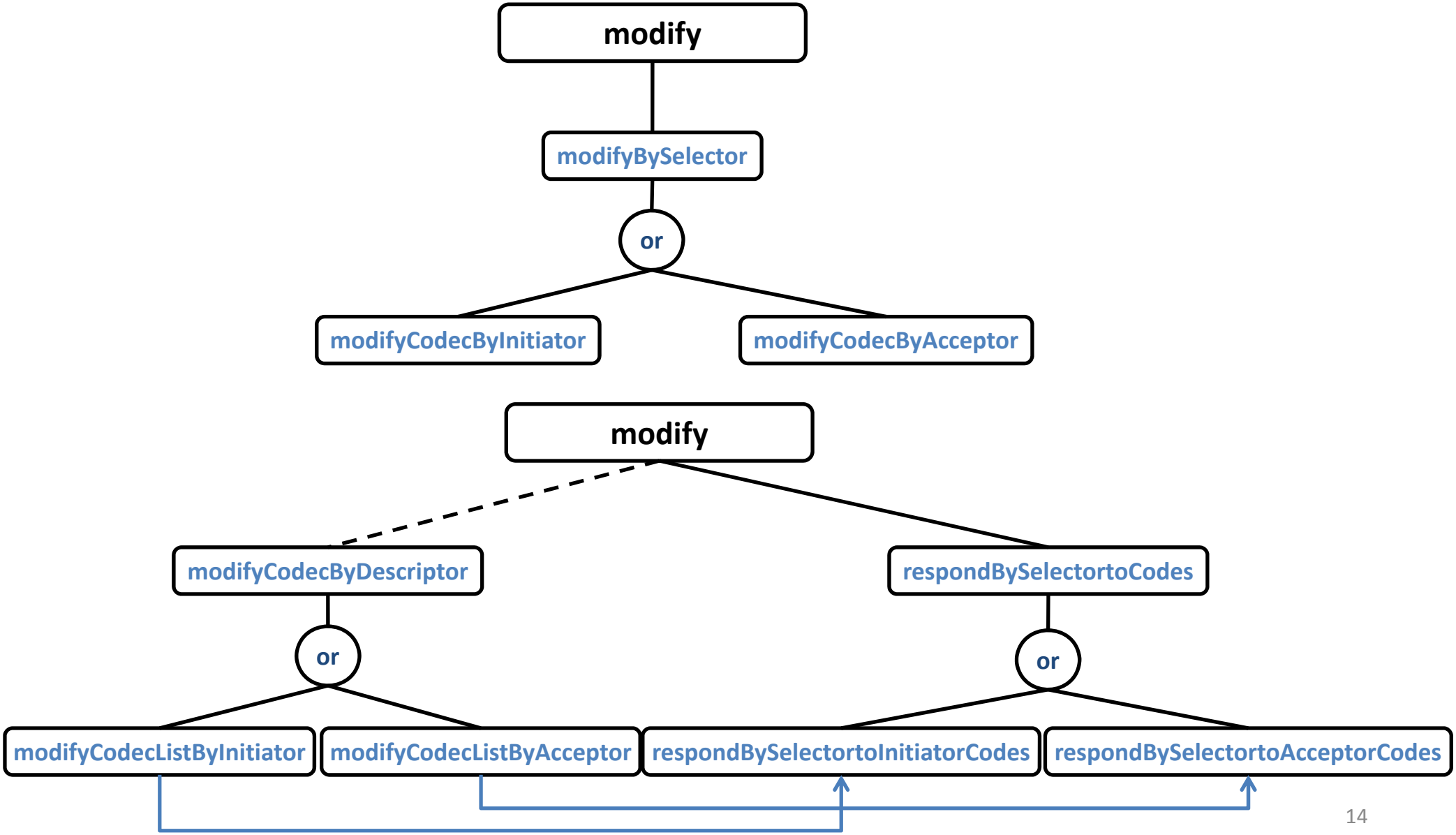
* Pamela Zave and Eric Cheung. Compositional Control of IP Media. IEEE Trans. Software Eng., 2009.

Multi Media Protocol, Close

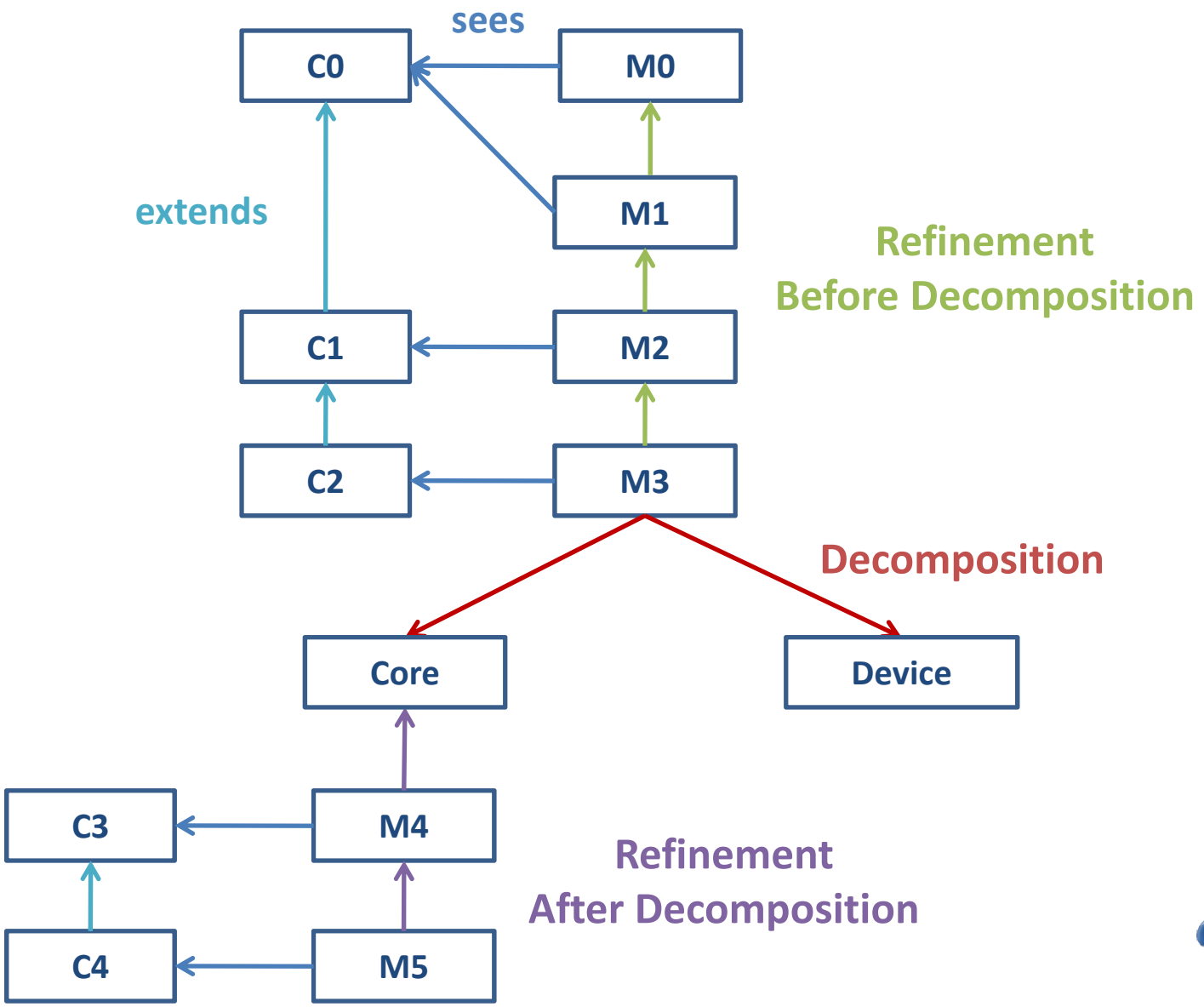


- Guard Lines (pre-condition)

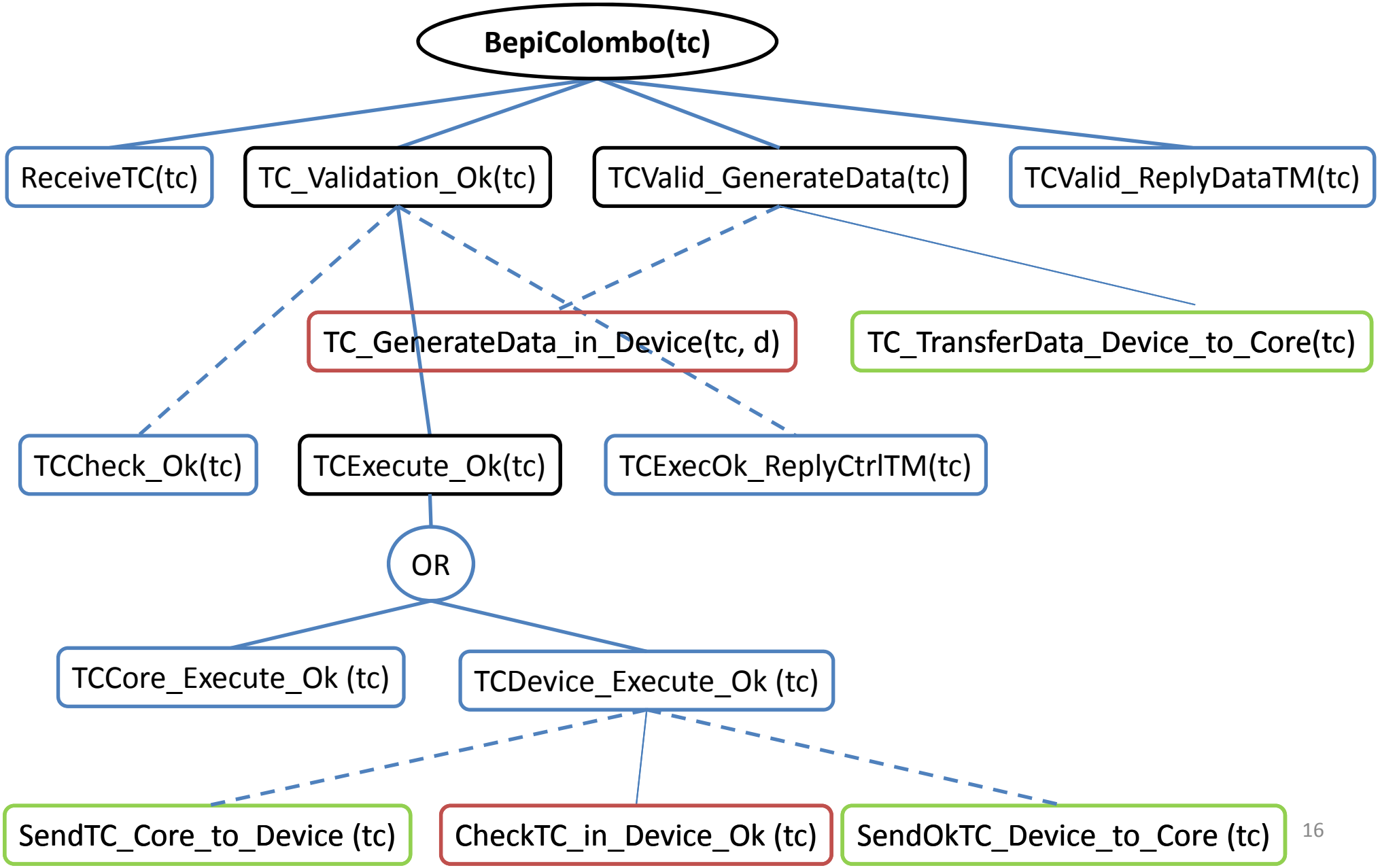
Multi Media Protocol, Modify



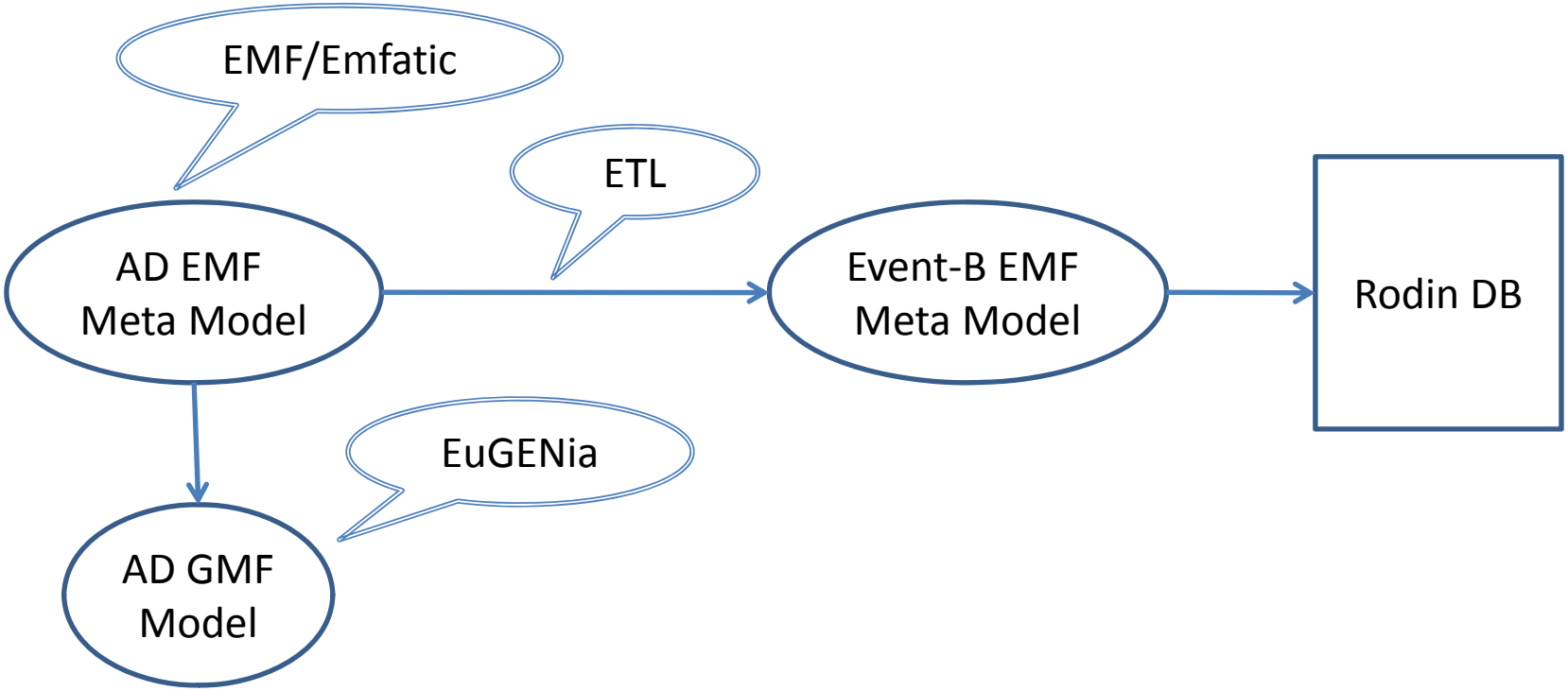
Space craft, Development Architecture



Space Craft, Atomicity Decomposition



Atomicity Decomposition Plug-in in Rodin Platform



- Automatic Event-B model generation
- Decrease modelling effort
- Reusability via patterns
- Overall view of the refinement process

Conclusion

- Overview of refinement process
- Visual refinement strategy
- Explicit events ordering in Event-B model
- Evaluation via couple of complex case studies
- Reusability via refining patterns and constructs
- Tool support and automatic model generation

The End

Questions ?

Thank you for your Attention