FROM RESEARCH TO INDUSTRY

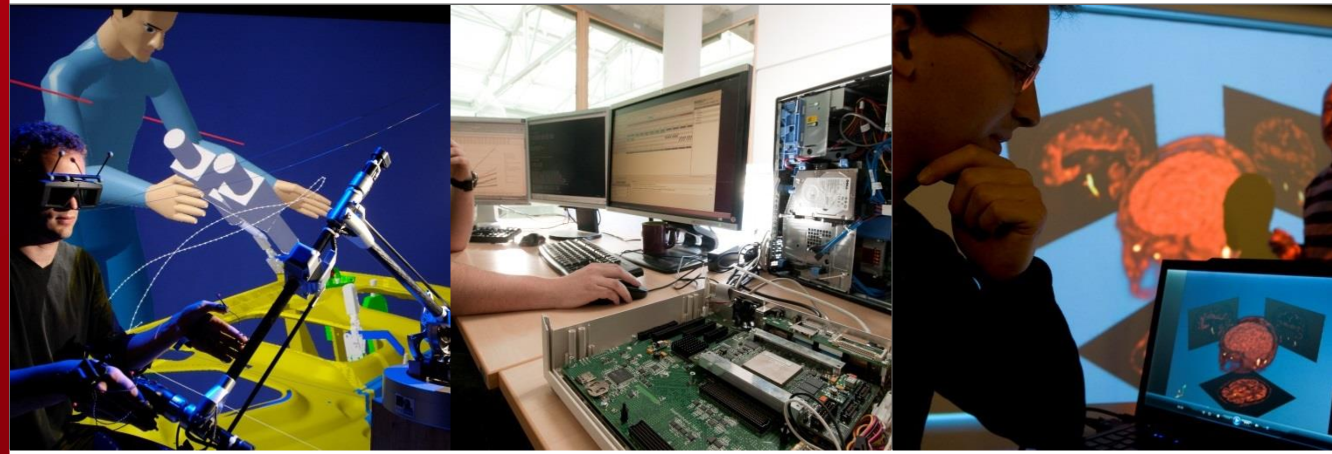
cea tech

PAPYRUS ROUND-TRIP ENGINEERING

Van Cam Pham, Shuai Li, Ansgar Radermacher, Cédric Dumoulin, Chokri Mraidha, Rémi Schnekenburger, François Le Fèvre, Sébastien Gérard

{first_name}.{last_name}@cea.fr

list



INSTITUT CARNOT CEA LIST

digiteo

papyrus

CONTENT

CONTENT

What is round-trip engineering?

"The ability to automatically maintain the consistency of multiple, changing software artifacts, in software development environments/tools, is commonly referred to as round-trip engineering" [Sendall04]

- **Related to traditional software engineering disciplines:**
  - Forward engineering: creating software from specifications
  - Reverse engineering: creating specifications from existing software
- **Round-trip engineering adds synchronization of existing artifacts that evolved concurrently by incrementally updating each artifact to reflect changes made to the other artifact**
- **Round-trip generalizes both forward and reverse engineering**

[Sendall04] S. Sendall and J. Küster. Taming model round-trip engineering. In *Proceedings of Workshop on Best Practices for Model-Driven Software Development*, Vancouver, Canada, 2004.

A real need for round-trip engineering

- Previously, UML-RT runtime code was manually reversed to a Papyrus UML model and code was generated
- The task was “long” and “tedious”
- In the mean time model and code evolved concurrently

! A real problem of synchronization

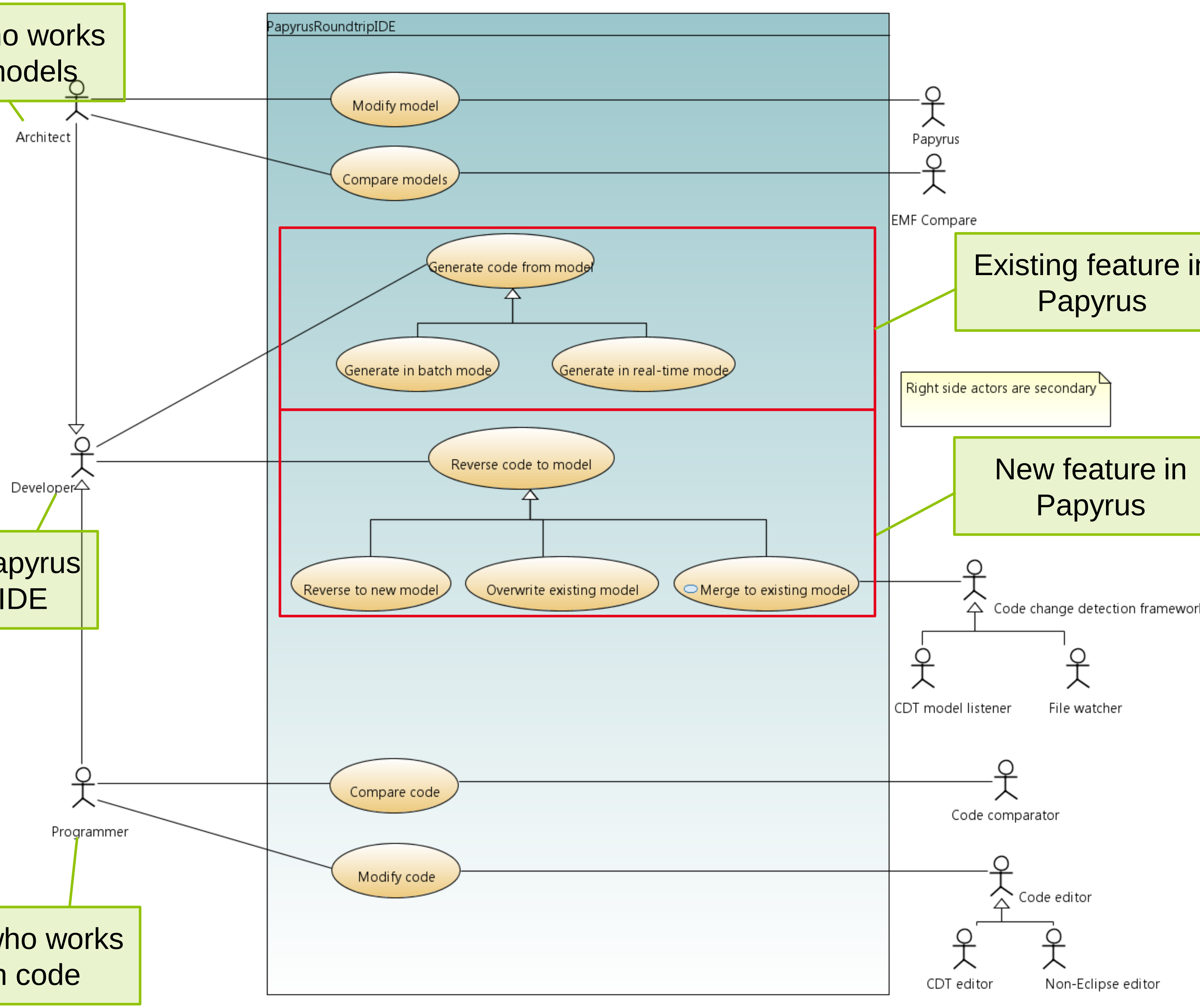
Papyrus offer for round-trip engineering

- UML modeler and C++ code generator are released…
- …but reverse and incremental update tool are still in development

Propose a IDE for **C++** round-trip engineering

ROUND-TRIP USE-CASES AND SCENARIOS

A developer who works mostly with models



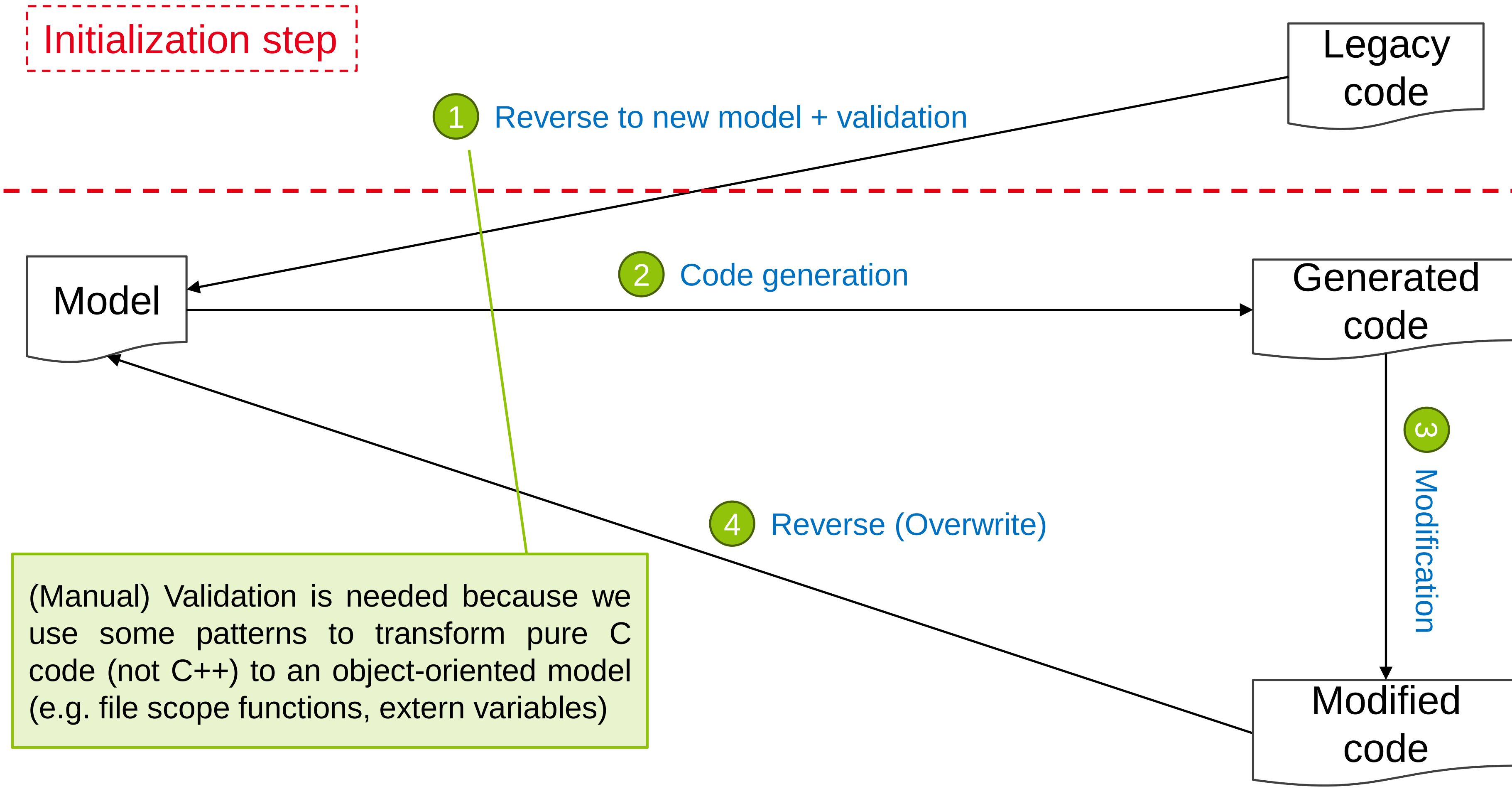
Existing feature in Papyrus

New feature in Papyrus

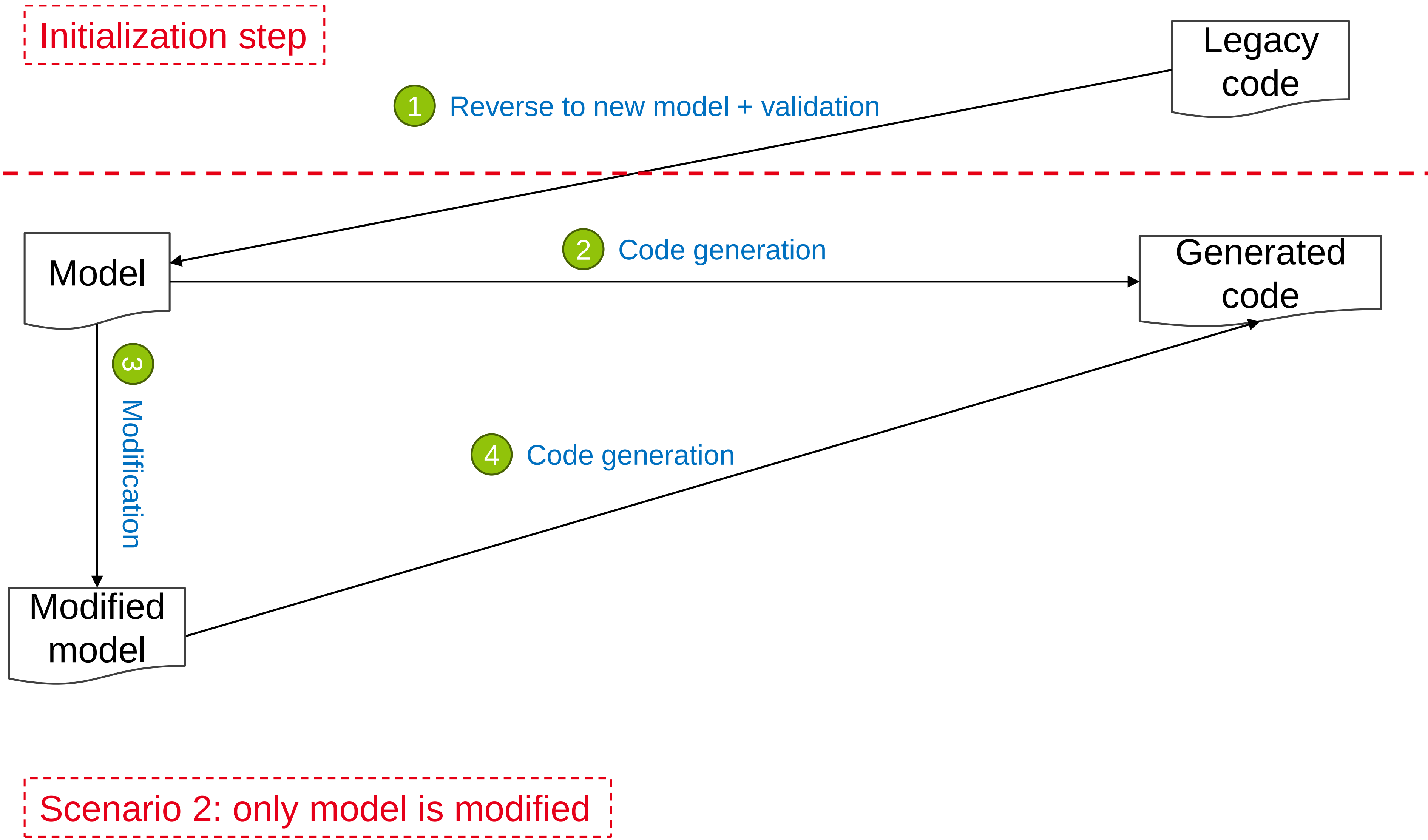
User of the Papyrus Round-trip IDE

A developer who works mostly with code

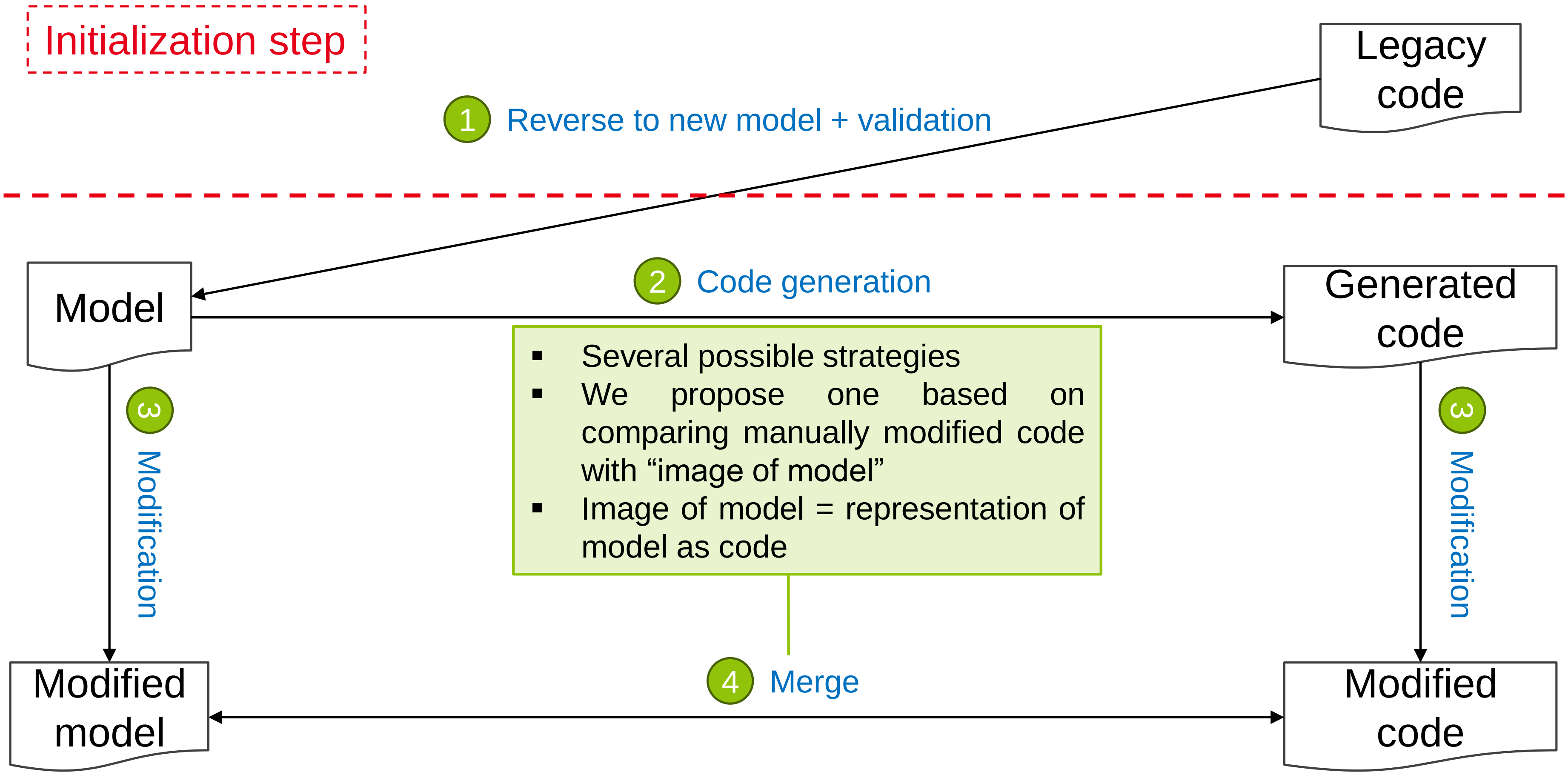
ROUND-TRIP USE-CASES AND SCENARIOS



Scenario 1: only code is modified


Initialization step
Legacy code
1 Reverse to new model + validation
Model
2 Code generation
Generated code
3 Modification
4 Code generation
Modified model
Scenario 2: only model is modified

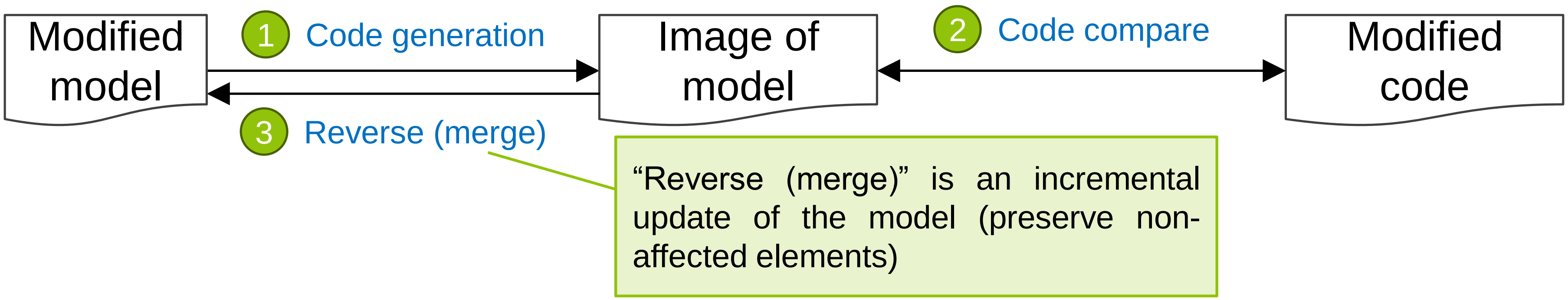
ROUND-TRIP USE-CASES AND SCENARIOS



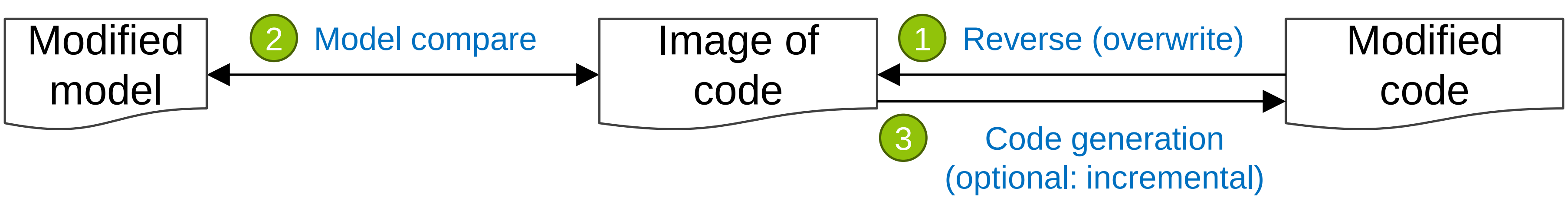
Scenario 3: model and code are both modified

ROUND-TRIP USE-CASES AND SCENARIOS

Scenario 3.1: merge by creating an image of the model as code

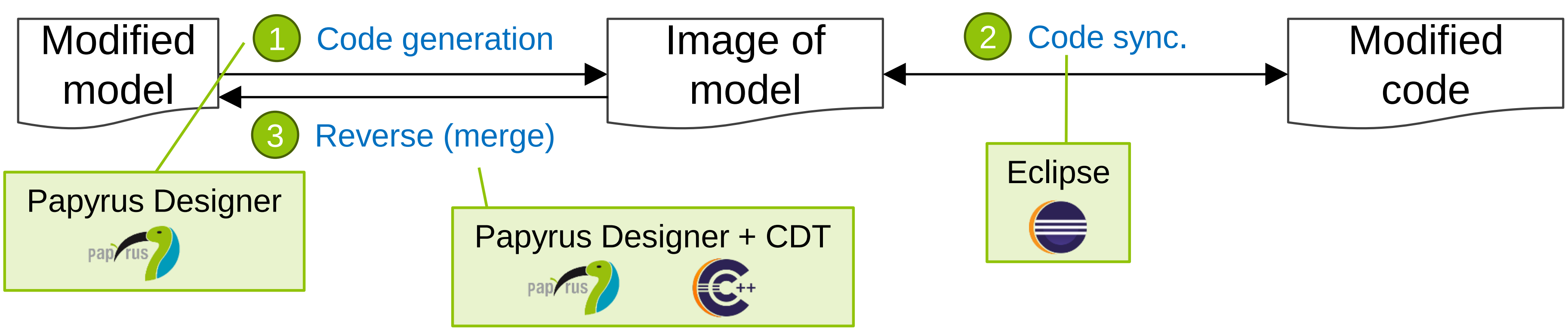


Scenario 3.2: merge by creating an image of the code as model

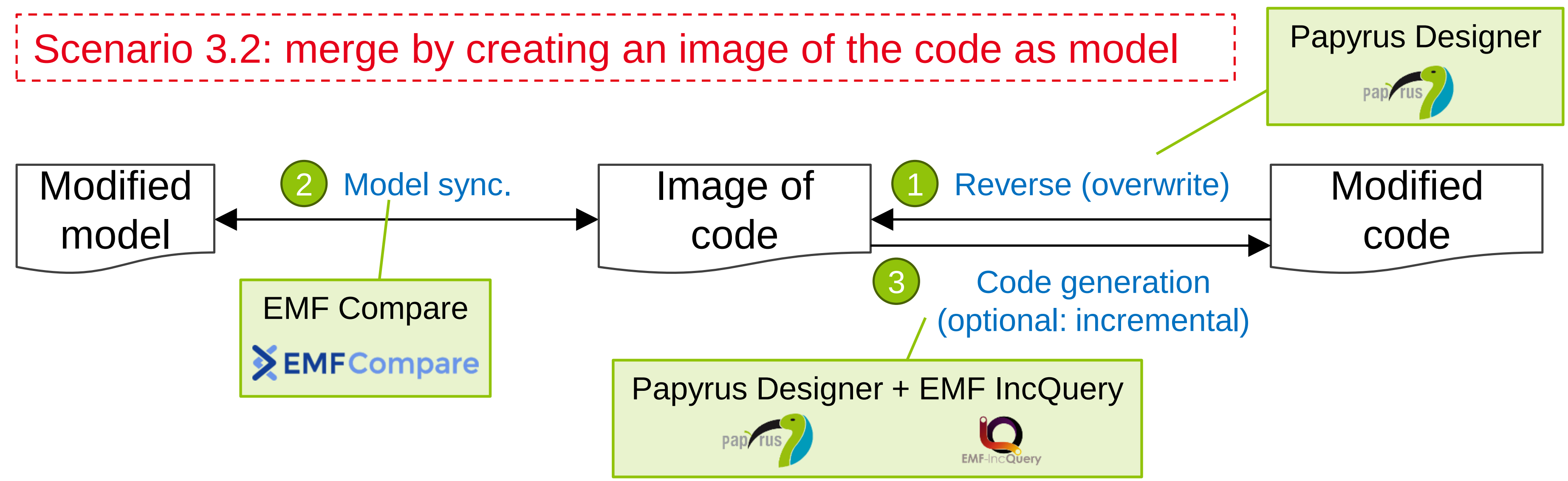


ROUND-TRIP USE-CASES AND SCENARIOS

Scenario 3.1: merge by creating an image of the model as code



Scenario 3.2: merge by creating an image of the code as model



Technology comparison

Criteria	Merge using code	Merge using model	Winner
UI usability	Suited for algorithmic changes	Suited for architectural changes	Draw
Change detector	CDT listener is unreliable and not fine-grained	EMF IncQuery is reliable and fine-grained	Merge using model
Comparison tool	Many mature comparators (but basic can be messy)	EMF Compare comparison fine-grained in Papyrus, needs some UI work outside of Papyrus, xmi:id limitation needs to be leveraged	Merge using code
Overall robustness	“Reverse (merge)” operation is dependent on code changes detector and handling of changes, which in the current state is error-prone	Incremental code generation is based on more reliable EMF IncQuery AND it’s optional	Merge using model
Technology dependency	CDT	EMF technologies	Draw

CONTENT

Video ▶

https://youtu.be/sudtoPvvyTA

CONTENT

Summary

- **Papyrus round-trip engineering proof of concept**
- **Most limitations are on the implementation side due to technology issues**

Current and future work

- **Unitary testing and debugging of reverse tool for robustness**
- **Improvement of code generator**
- **Test on several real case studies:**
  - UML-RT runtime
  - Diversity
  - Embedded applications developed by CEA in C++
  - LEGO EV3 C++ library
- **Java version (integration of Cédric Dumoulin's work)**
- **Research work (thesis of Van Cam Pham): incremental code generation**

Roadmap and project

- **Target Eclipse Neon M6 milestone (end Q1 2016) for release of round-trip features**
- **European project on this subject being proposed?**