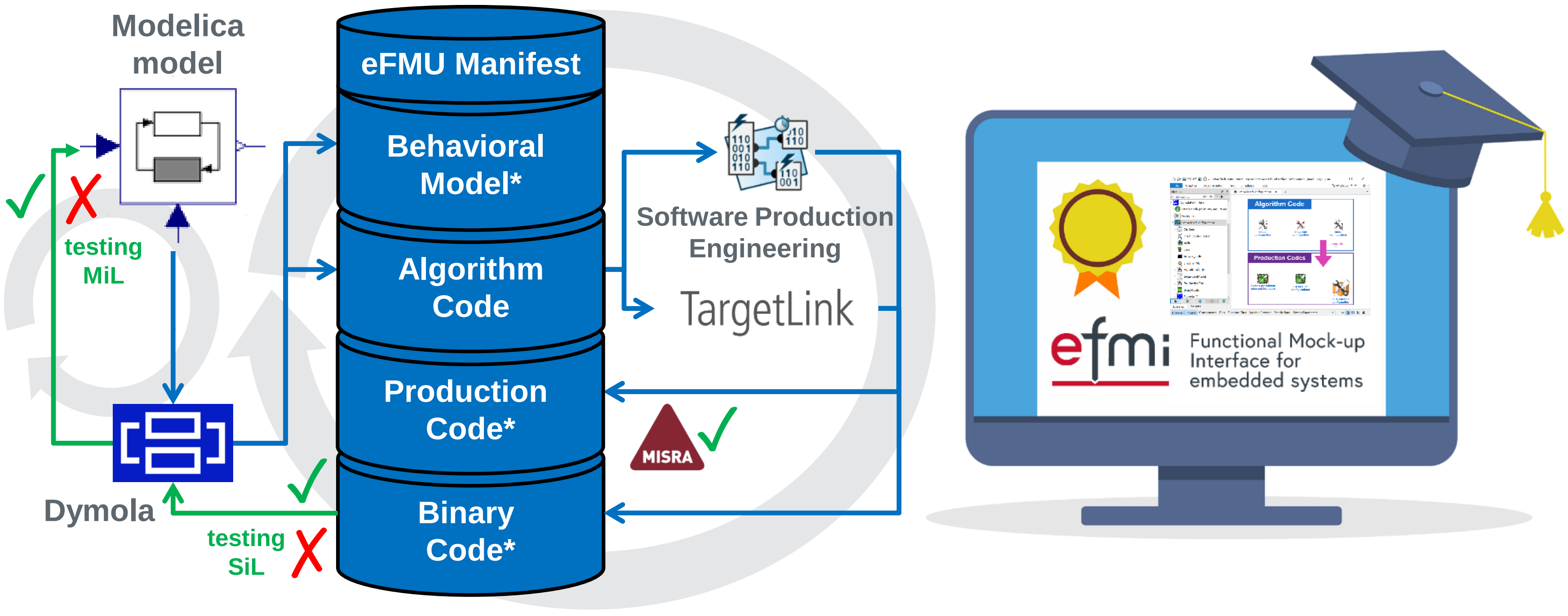


Part 1: eFMI® motivation and overview

eFMI®: A beginner’s overview and hands-on

– 16th International Modelica Conference – 8th of September 2025 –

Christoff Bürger
Dassault Systèmes
Christoff.Buerger@3ds.com

eFMI® tutorial – Agenda

Part 1: eFMI® motivation and overview (40 min)

Part 2: Running use-case introduction (10 min)

Part 3: Hands-on in Dymola and Software Production Engineering (25 min)

Coffee break (30 min)

Part 3: Hands-on in Dymola and Software Production Engineering (30 min)

Part 4: Advanced demonstrators (20 min)

Part 5 (industry case-study): eFMI based thermal management system (TMS) development for fuel cell electric vehicles (FCEV) (20 min)

Part 6: Outlook and conclusion (5 min)

Tutorial leader:
Christoff Bürger

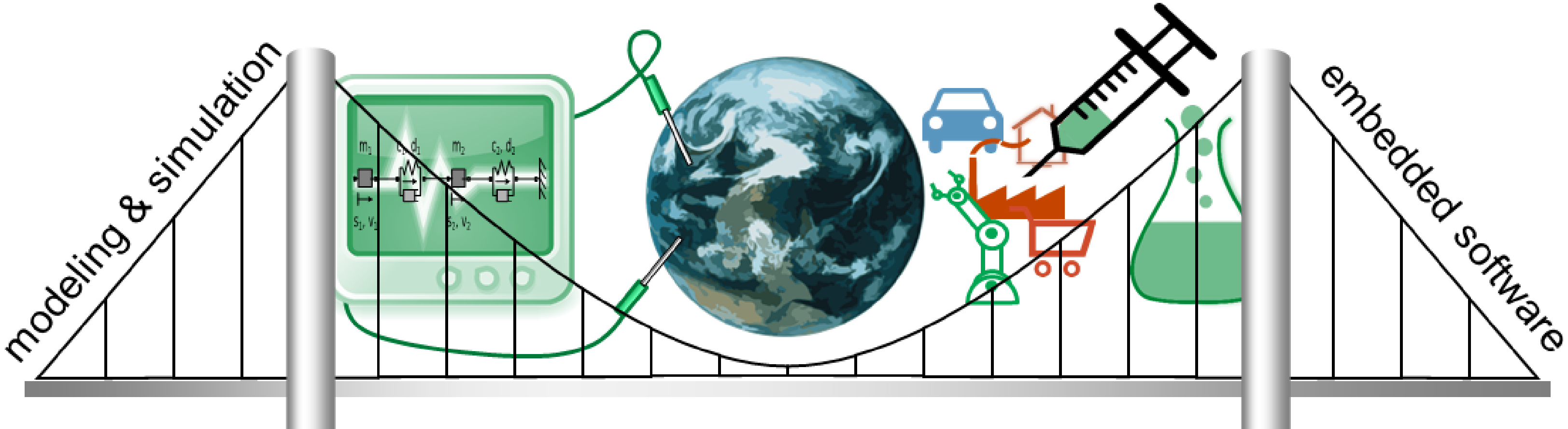
DASSAULT SYSTEMES

Presenter:
Daeoh Kang

iVH Institute of Vehicle Engineering

efmi Functional Mock-up Interface for embedded systems

Modelica Association Project eFMI (MAP eFMI)



efmi Functional Mock-up Interface for embedded systems

Modelon
ALTAIR
iVH Institute of Vehicle Engineering
dSPACE
DLR
DASSAULT SYSTEMES
MITSUBISHI ELECTRIC Changes for the Better
BOSCH Invented for life
SIEMENS Ingenuity for life
EFS ELEKTRONISCHE FAHRWERKSYSTEME
OpenModelica
ETAS DRIVING EMBEDDED EXCELLENCE
KEYSIGHT TECHNOLOGIES
Mercedes-Benz

**Project leader:
Christoff Bürger**

DASSAULT SYSTEMES

**Deputy project leader:
Hubertus Tummescheit**

THE REUSE COMPANY

https://efmi-standard.org/

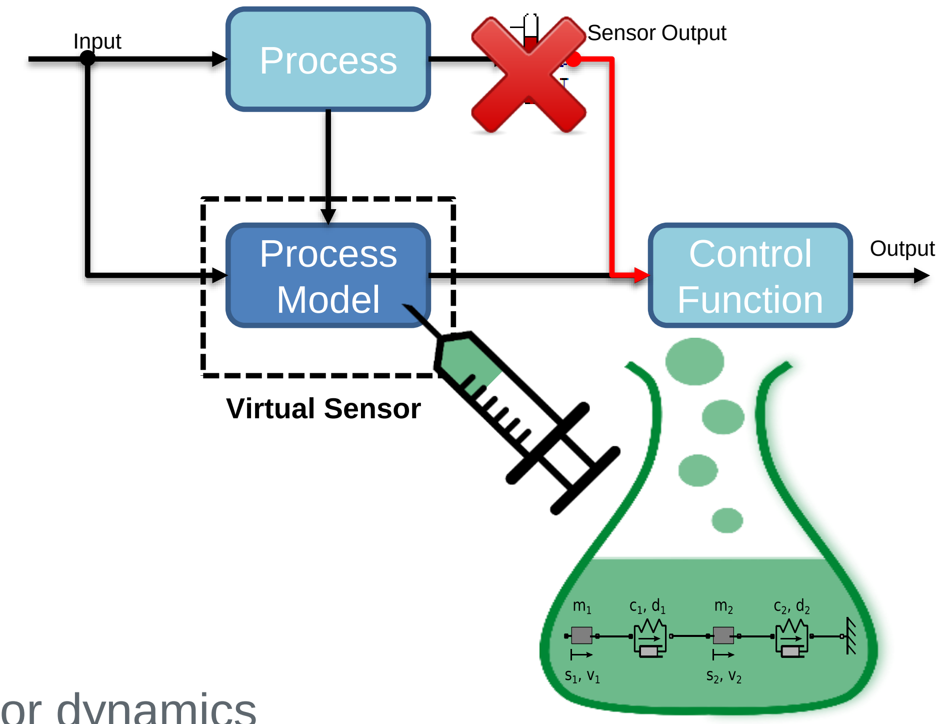
What is it all about?

Functional Mock-up Interface for embedded systems

eFMI motivation: Advanced control is challenging

Online physics models key technology for advanced (engine) control software:

- Virtual sensors, i.e., observers,
- Model-based diagnosis
- Inverse physical models as feed forward part of control structures
- Model predictive control



Physics model

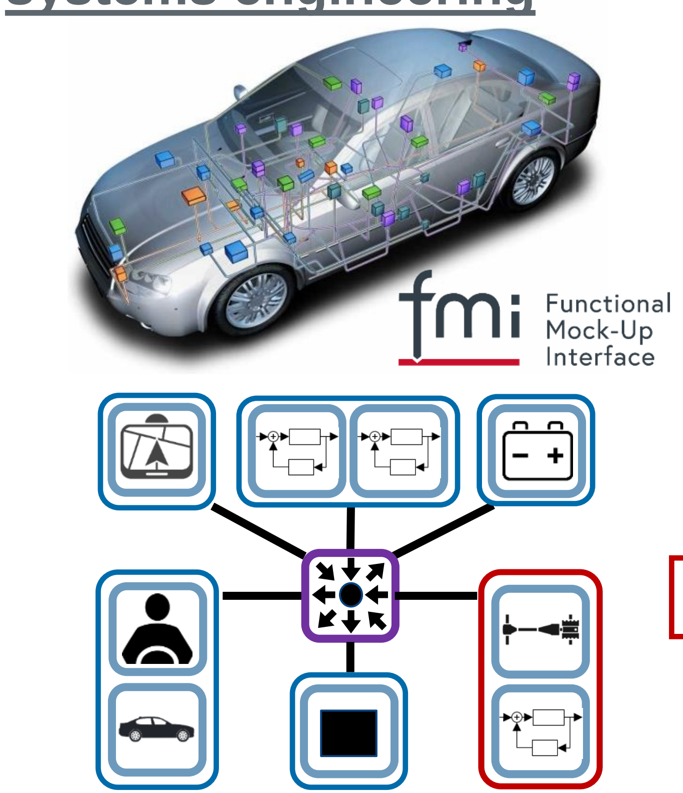
Physics models:

- Typically described by differential equations, best suited for dynamics
- Complementary to data-based modeling, can be combined
- Reduced calibration effort due to physical parameters

efmi Functional Mock-up Interface for embedded systems

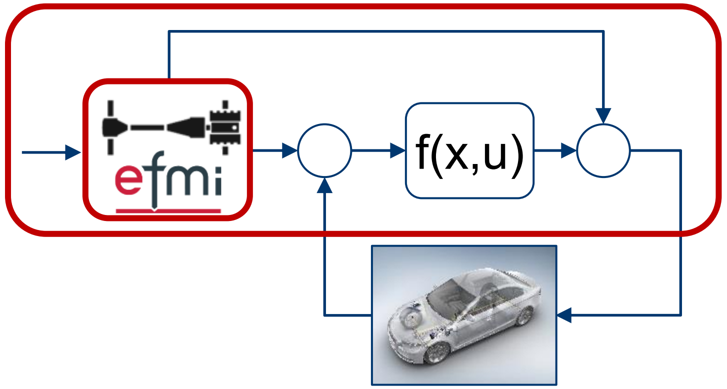
eFMI Standard: How it is different compared with FMI

Model-based systems engineering



Model-based control

- Virtual sensors
- Feed-forward control
- Model-based diagnosis
- Model predictive control
- Advanced operating strategies
- …



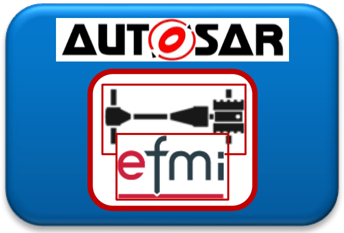
Software engineering

AUTOSAR
Application Layer (ASW)

Run Time Environment (RTE)

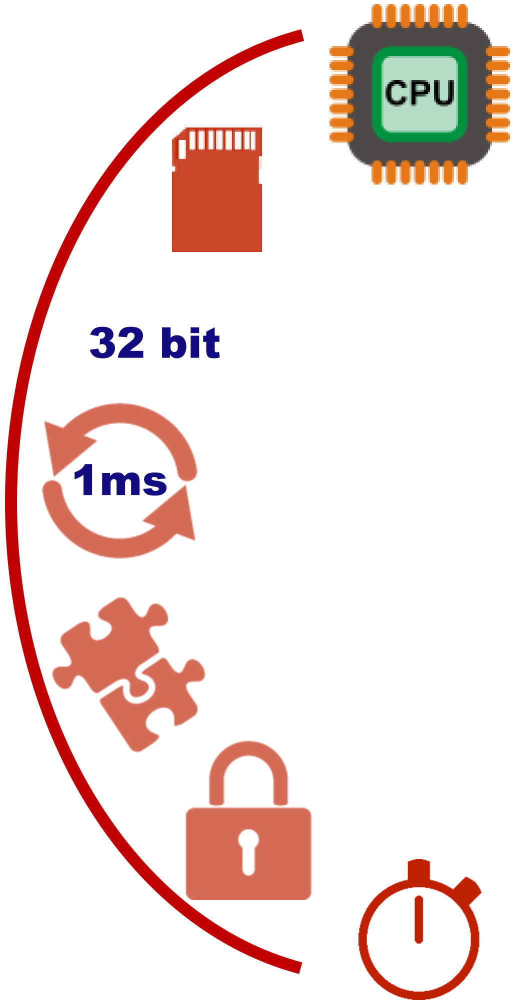
AUTOSAR Basic Software (BSW)

ECU Hardware



SW-C

eFMI motivation: Embedded systems are challenging



- Limited computation power
- Limited memory
- Limited precision
- Limited sampling rate
- Static memory allocation
- Guaranteed execution time
- Inbound guarantees
- No exceptions guarantees

- Specialized hardware

Bosch MDG1 ECU

- SW architectures

AUTOSAR

- Rules & regulations

MISRA

Motor **I**ndustry **S**oftware **R**eliability **A**ssociation

efmi Functional Mock-up Interface for embedded systems

eFMI motivation: Multi-domain collaboration is challenging

Control engineering

(system theory, stability, robustness, …)

Numerics

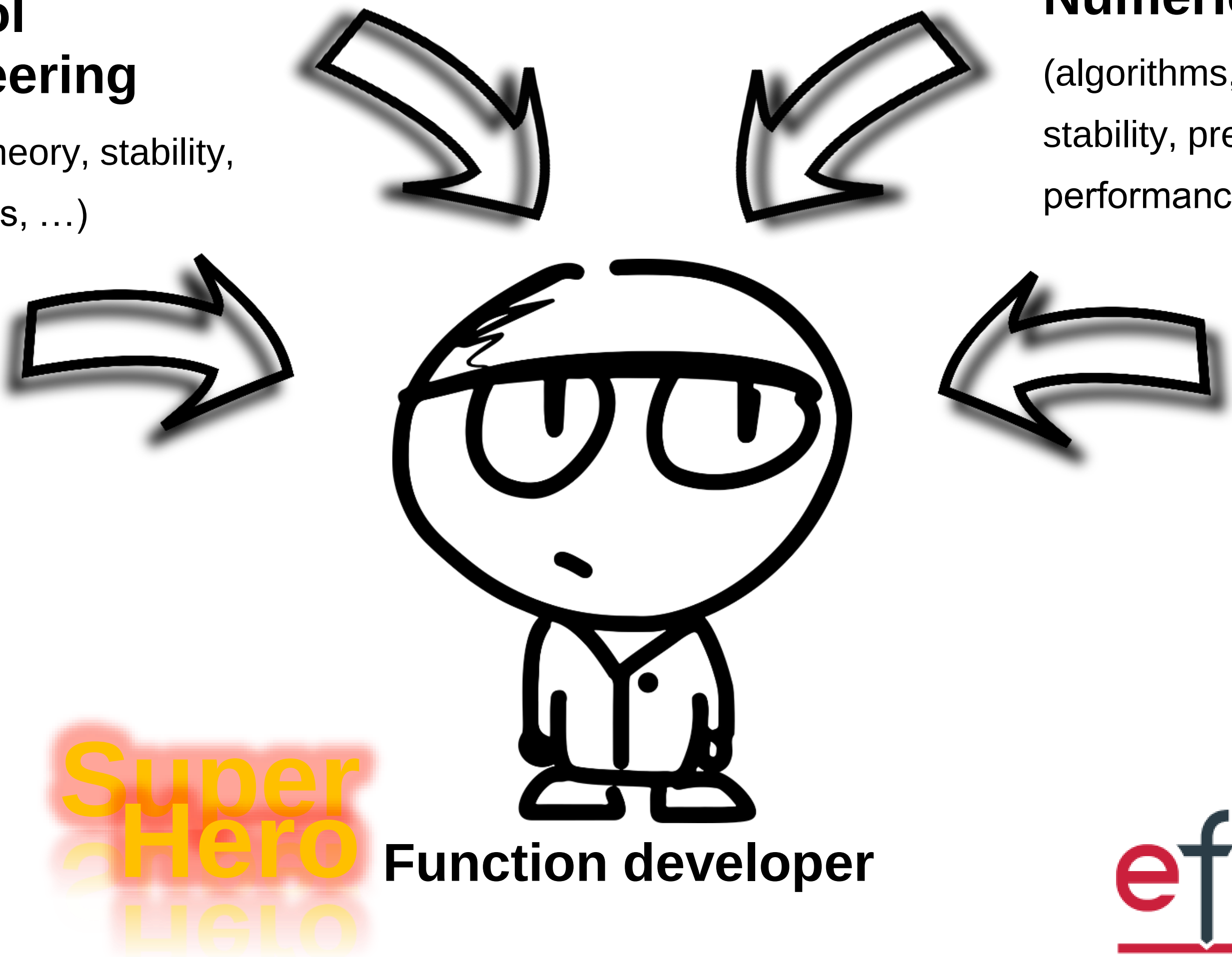
(algorithms, complexity, stability, precision, realtime performance…)

Physics modeling

(domain knowledge, physical principles & phenomena, system dynamics, model validation, …)

ECU software

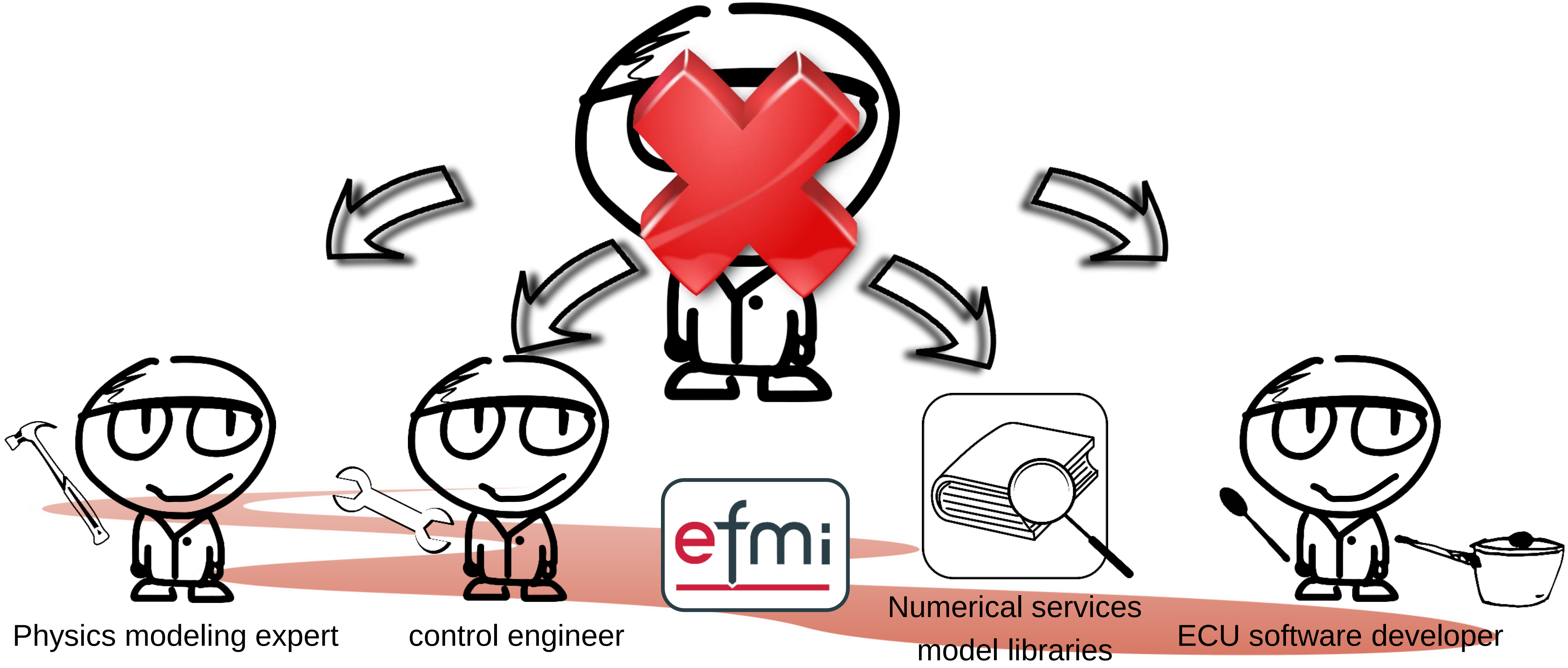
(MISRA, ASIL, MSR, AUTOSAR, …)



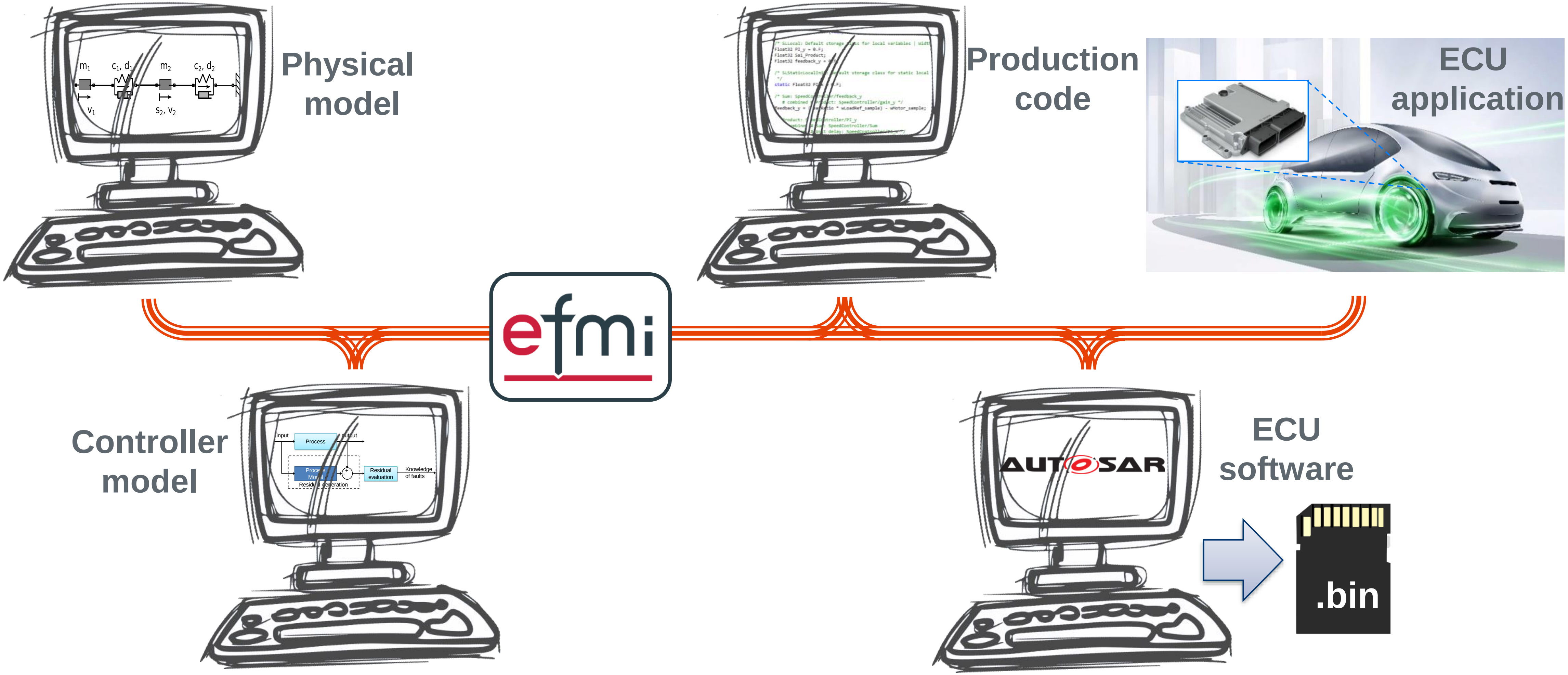
Function developer

efmi Functional Mock-up Interface for embedded systems

eFMI solution: Domain experts with dedicated tools…



eFMI solution: …automatizing a distinct development process…



Physical model

Production code

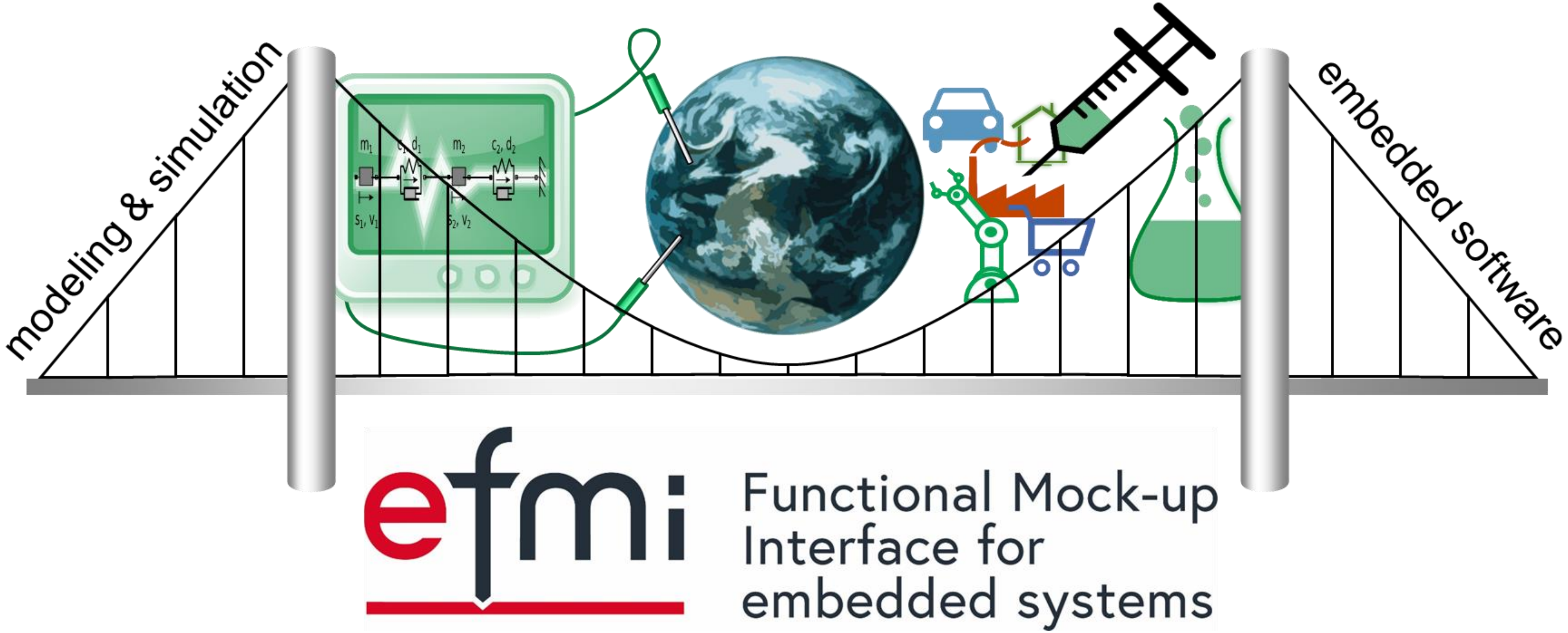
ECU application

Controller model

ECU software

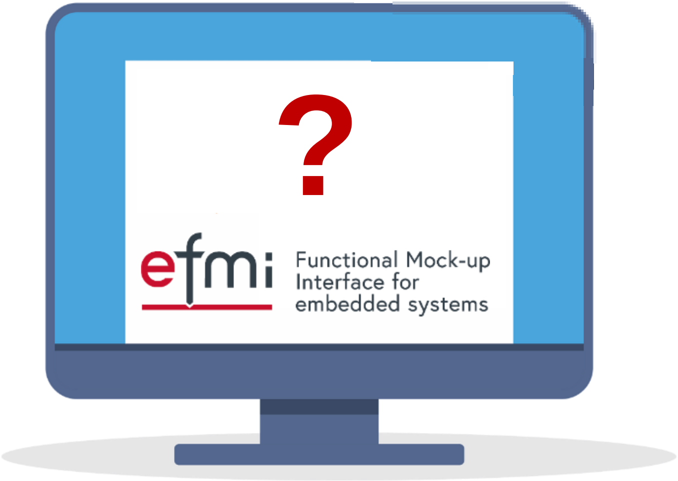
```
/* SLLocal: Default storage class for local variables | Width
Float32 PI_y = 0.F;
Float32 Sal_Product;
Float32 feedback_y = 0.F;

/* SLStaticLocalInit: Default storage class for static local
 */
static Float32 PI_x = 0.F;

/* Sum: SpeedController/feedback_y
   # combined # Product: SpeedController/gain_y */
feedback_y = (GearRatio * wLoadRef_sample) - wMotor_sample;

   Product: SpeedController/PI_y
     combined # Sum: SpeedController/Sum
         # Unit delay: SpeedController/PI_x */
```

eFMI solution: …defined by a common standard



Open standard for model-driven development of advanced control functions for safety-critical and real-time targets.

Ok, eFMI is about bringing physics simulations to safety-critical real-time targets!



But what *is* the eFMI Standard?

eFMI Standard: Mission

New standard enabling the application of (physics) models in embedded software:

- Workspace for step-wise development and refinement

  (from first high-level algorithmic solution to an embedded implementation on a dedicated target environment)

- Cover the development concerns of implementation, testing, and integration

eFMUs model representations support:

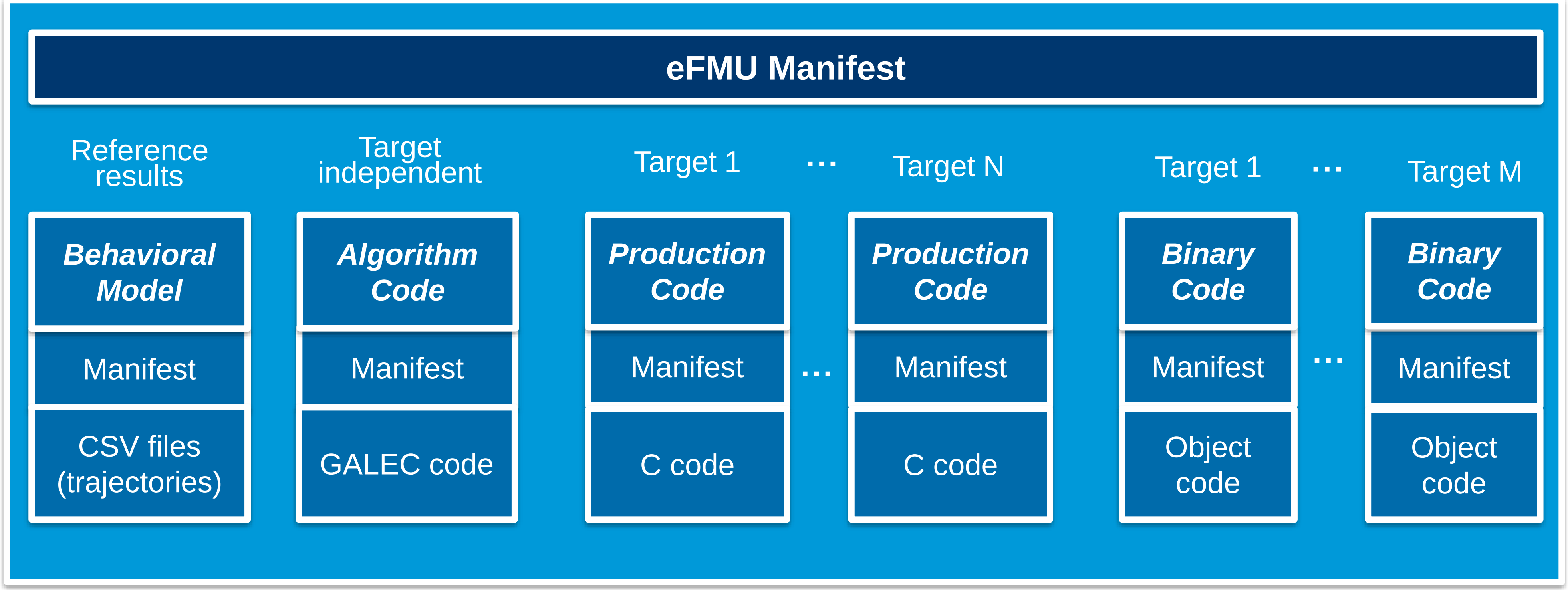
- *Behavioral Model* container: Behavior / reference results for testing.
- *Algorithm Code* container: Target-independent bounded algorithmic solution based on *GALEC*

  (new programming language for safety-critical, real-time suited, fix-rate sampled algorithms)

- *Production Code* container: C implementations, tailored and optimized for target environment requirements
- *Binary Code* container: Binary distributions and their „build-recipes“, ready for embedded system integration

efmi Functional Mock-up Interface for embedded systems

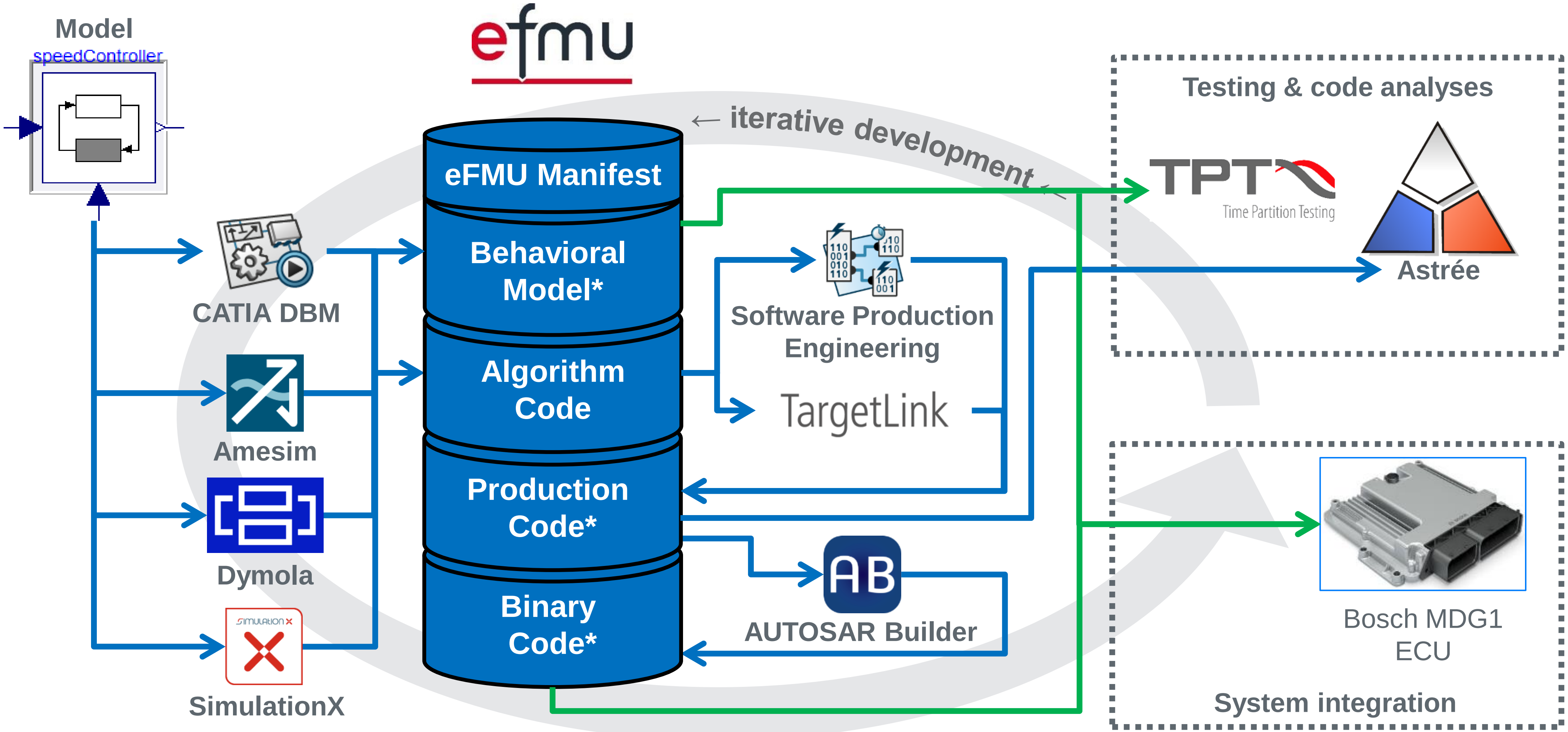
eFMI Standard: Container architecture



efmu

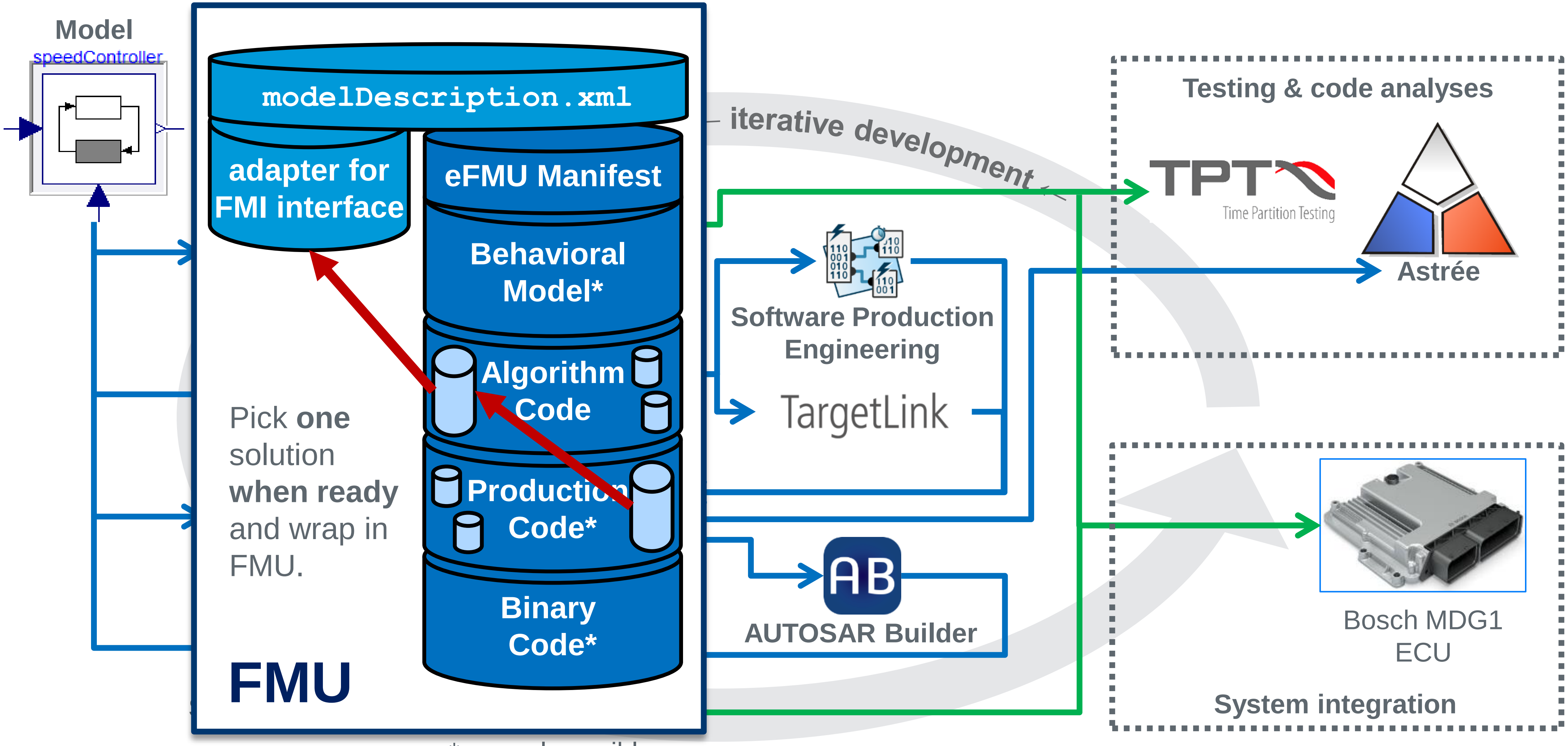
efmi Functional Mock-up Interface for embedded systems

eFMI Standard: Toolchain & workflow



*several possible

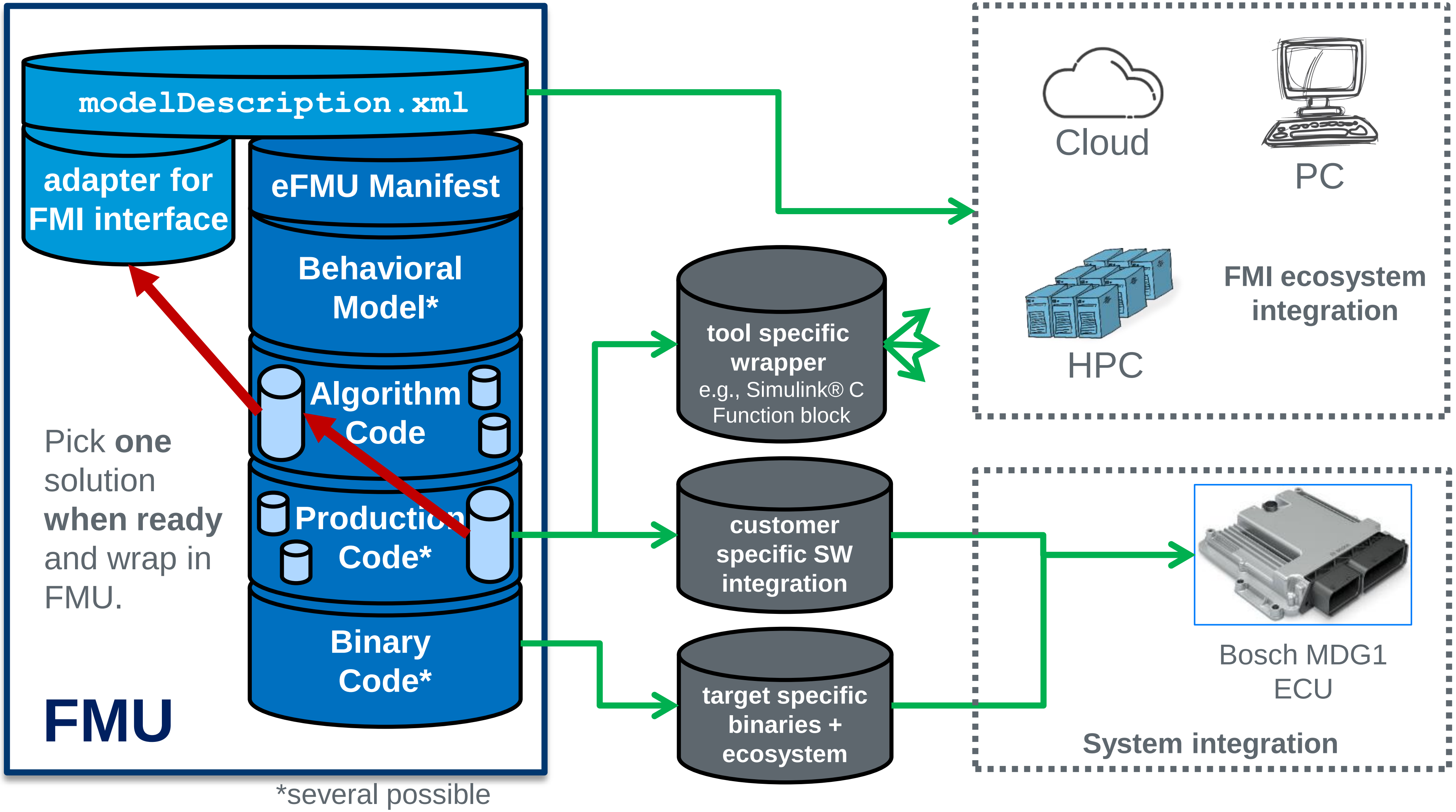
eFMI Standard: Deployment scenarios



*several possible

eFMI Standard: Deployment scenarios

Use existing standards / ecosystems for system integration (**not defined by eFMI**).


modelDescription.xml
adapter for FMI interface
eFMU Manifest
Behavioral Model*
Algorithm Code
Production Code*
Binary Code*
FMU
Pick **one** solution **when ready** and wrap in FMU.
tool specific wrapper
e.g., Simulink® C Function block
customer specific SW integration
target specific binaries + ecosystem
Cloud
PC
HPC
FMI ecosystem integration
Bosch MDG1 ECU
System integration

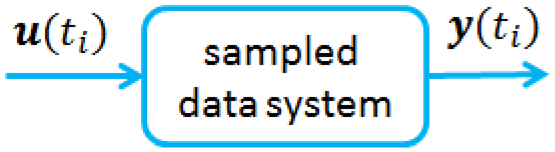
*several possible

Ok, the eFMI Standard defines model representations capturing embedded software development stages, leaving finding solutions to expert tools!

But how does the eFMI Standard enable an automatized toolchain satisfying functional, safety-critical and real-time objectives?

eFMI Standard: Container architecture & traceability

GALEC program:

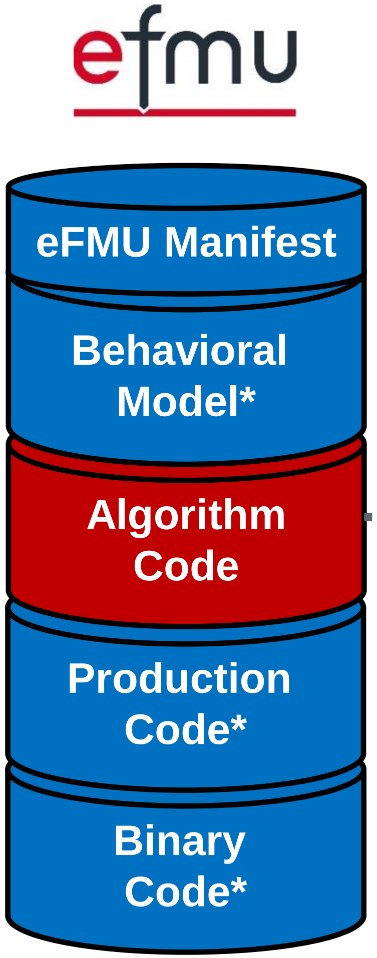


$$x_{i+1} = f_x(x_i, u_i)$$
$$y_i = f_y(x_i, u_i)$$

GALEC language features:

- Imperative, target independent with high math-abstraction level
- Well-defined, decidable semantics and safe numerics
- Guaranteed error handling
- Simple; high potential for target code tailoring & optimization

⇒ Nice intermediate representation for code generation (modelling target & embedded source)



*several possible

GALEC program

```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

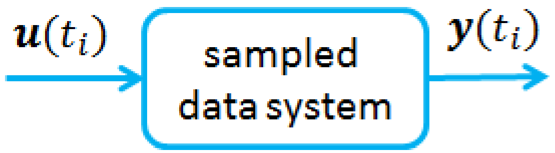
GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability

GALEC program:

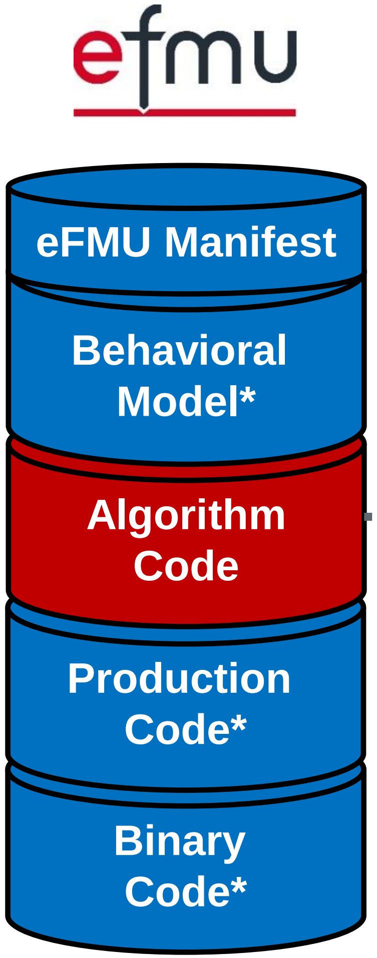


$$x_{i+1} = f_x(x_i, u_i)$$
$$y_i = f_y(x_i, u_i)$$

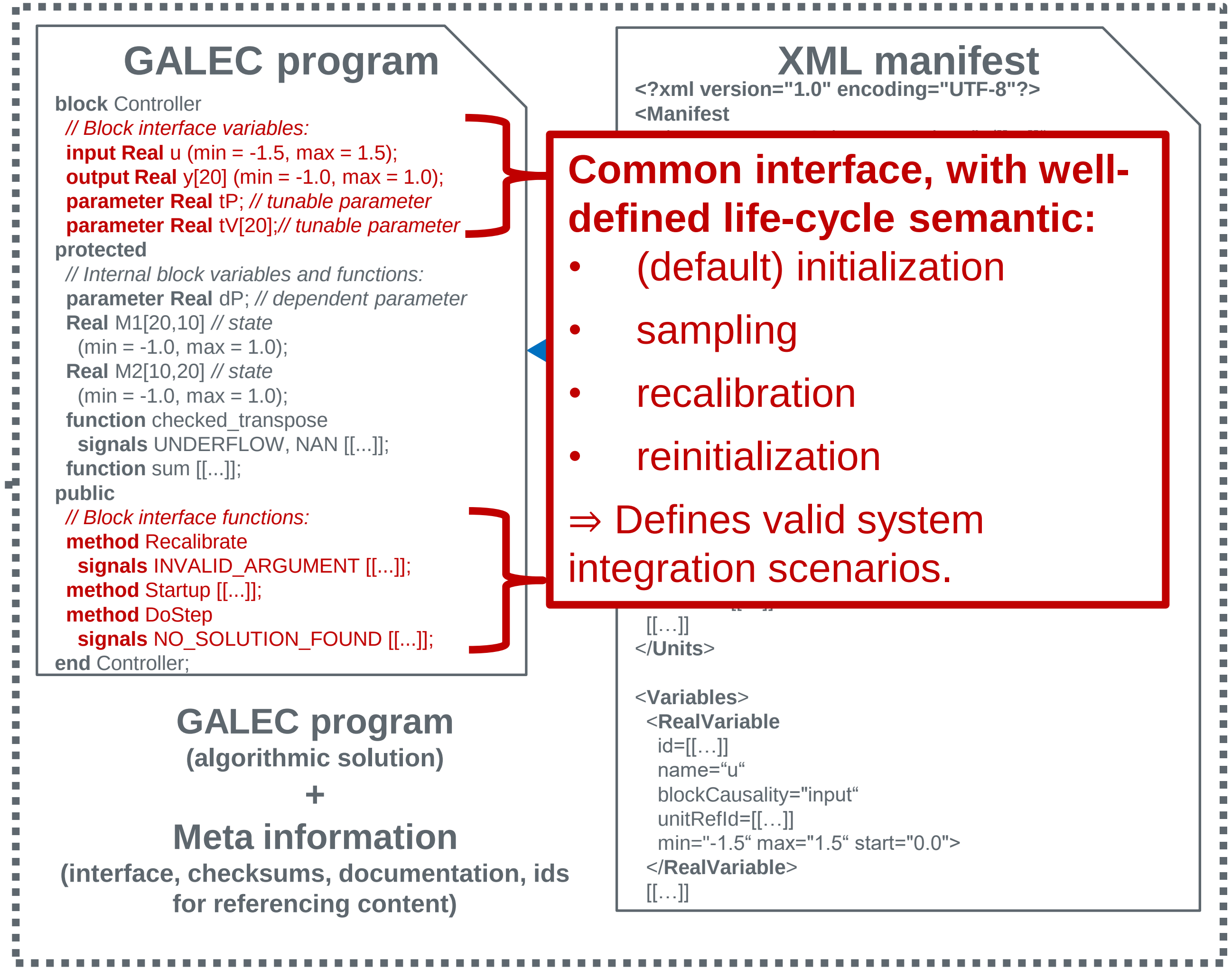
GALEC language features:

- Imperative, target independent with high math-abstraction level
- Well-defined, decidable semantics and safe numerics
- Guaranteed error handling
- Simple; high potential for target code tailoring & optimization

⇒ Nice intermediate representation for code generation (modelling target & embedded source)



*several possible



GALEC program

```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

Common interface, with well-defined life-cycle semantic:

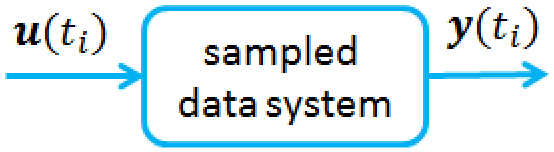
- (default) initialization
- sampling
- recalibration
- reinitialization

⇒ Defines valid system integration scenarios.

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

eFMI Standard: Container architecture & traceability

GALEC program:

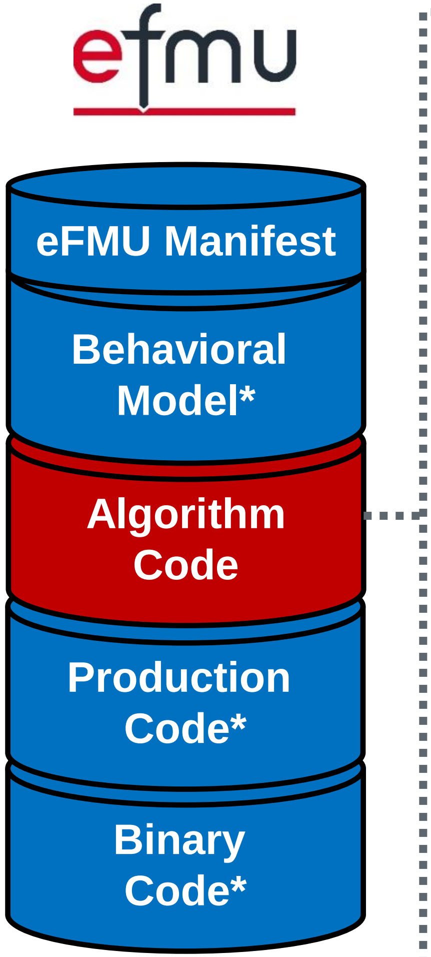


$$x_{i+1} = f_x(x_i, u_i)$$
$$y_i = f_y(x_i, u_i)$$

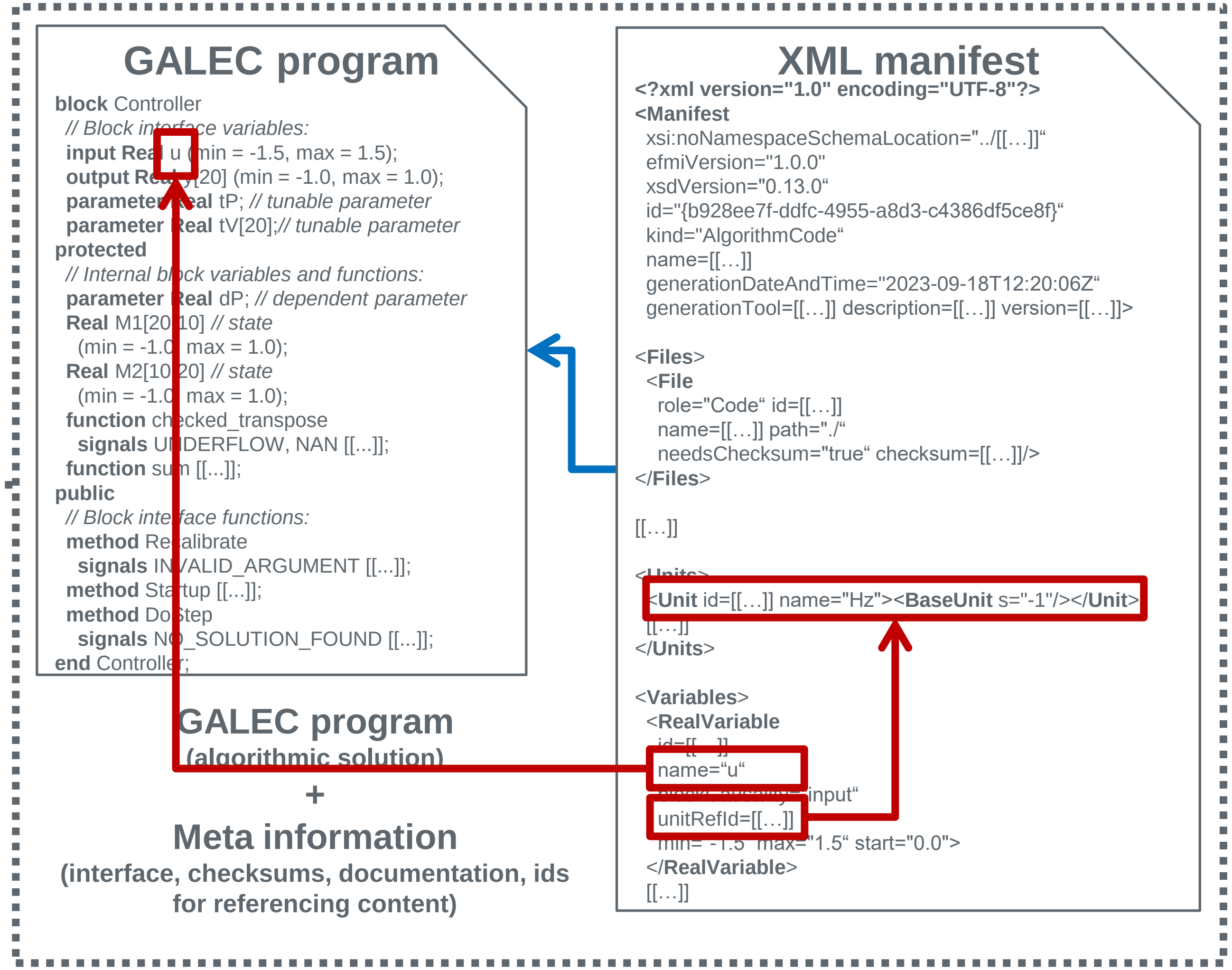
GALEC language features:

- Imperative, target independent with high math-abstraction level
- Well-defined, decidable semantics and safe numerics
- Guaranteed error handling
- Simple; high potential for target code tailoring & optimization

⇒ Nice intermediate representation for code generation (modelling target & embedded source)



*several possible



GALEC program

```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

XML manifest

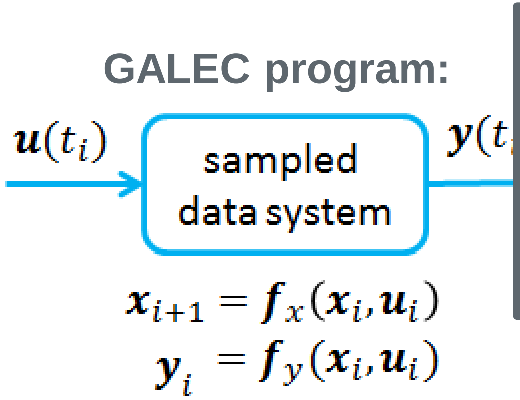
```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5" max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

GALEC program (algorithmic solution)

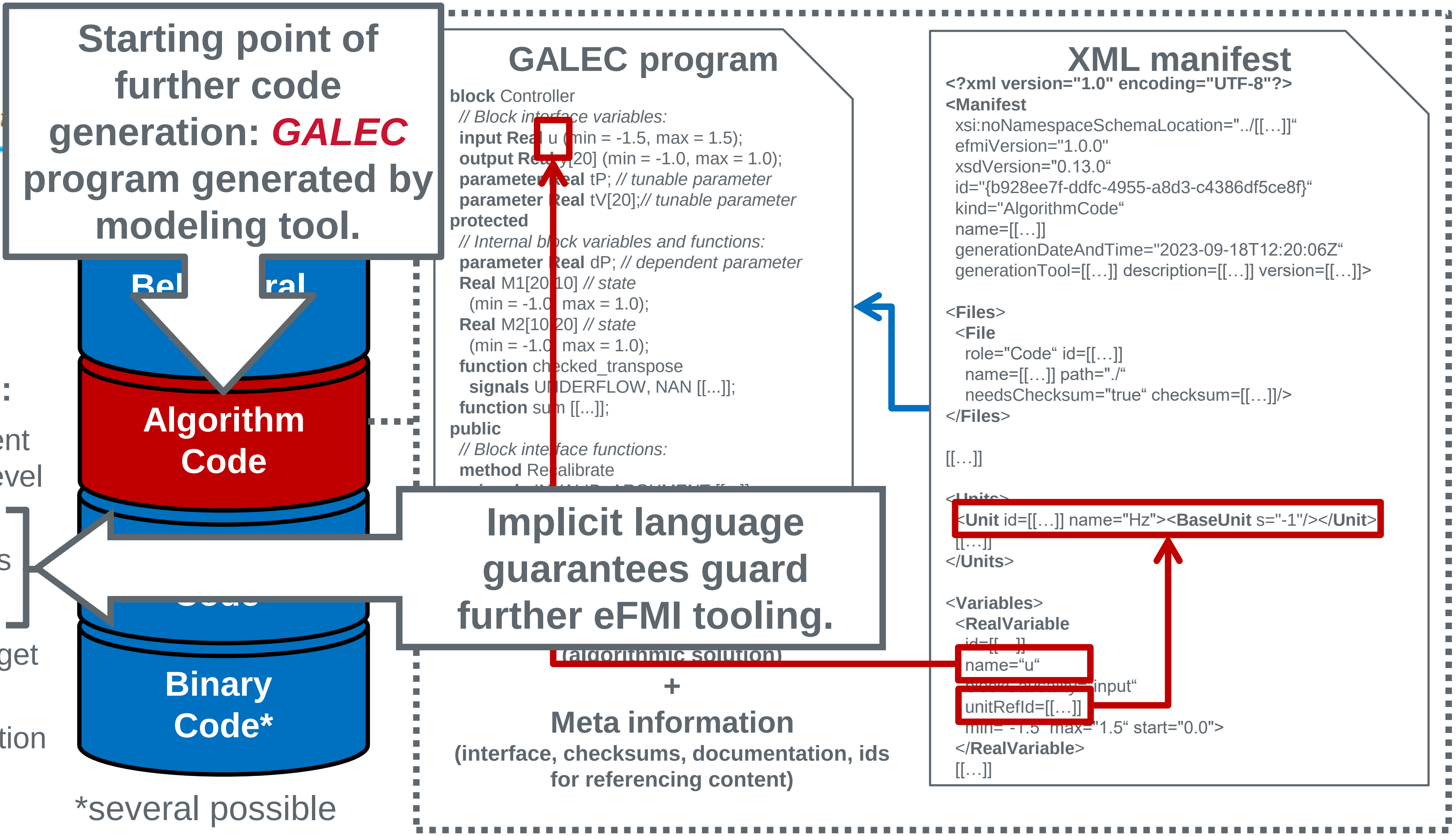
+

Meta information (interface, checksums, documentation, ids for referencing content)

eFMI Standard: Container architecture & traceability



$$x_{i+1} = f_x(x_i, u_i)$$
$$y_i = f_y(x_i, u_i)$$


Starting point of further code generation: GALEC program generated by modeling tool.
Algorithm Code
Binary Code*
Implicit language guarantees guard further eFMI tooling.
GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)
XML manifest

```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
```

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    input“
    unitRefId=[[…]]
    min="-1.5" max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

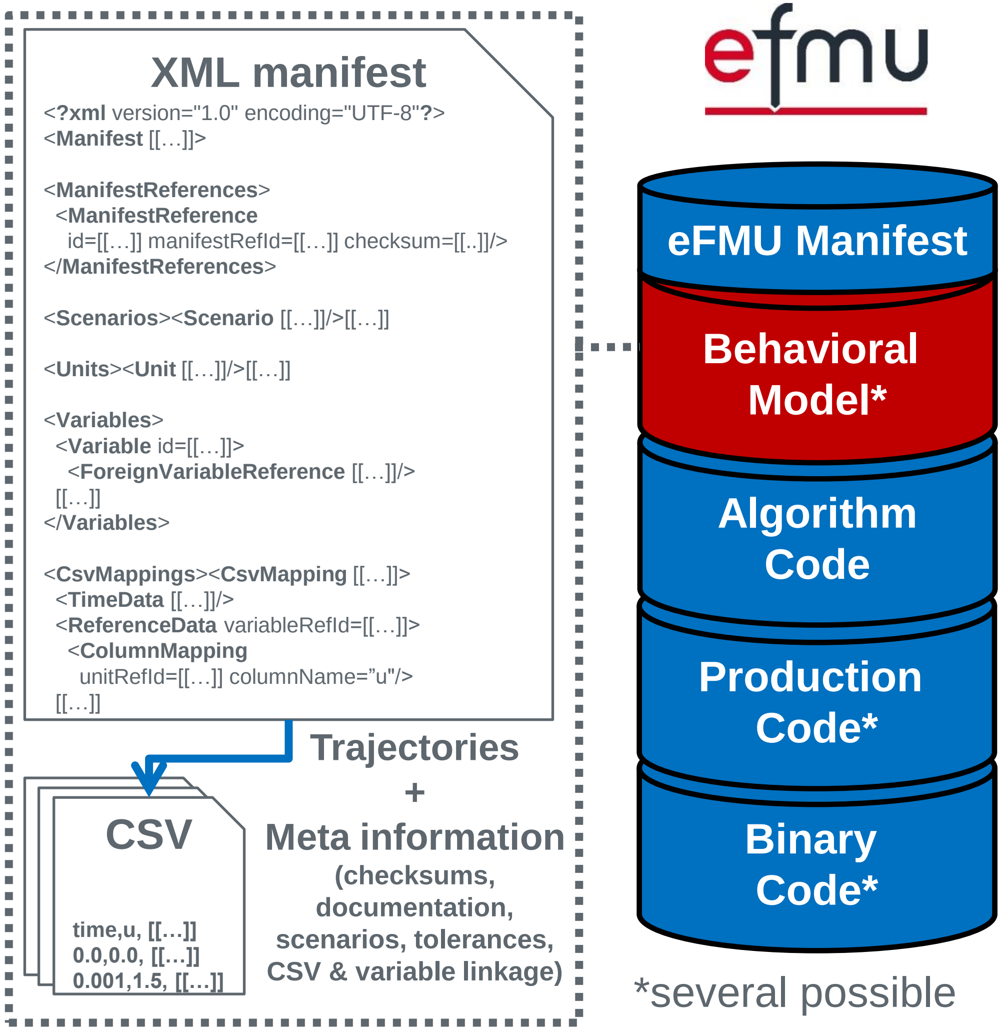
*several possible

GALEC language features:

- Imperative, target independent with high math-abstraction level
- Well-defined, decidable semantics and safe numerics
- Guaranteed error handling
- Simple; high potential for target code tailoring & optimization

⇒ Nice intermediate representation for code generation (modeling target & embedded source)

eFMI Standard: Container architecture & traceability



```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
  <ManifestReference
    id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Scenarios><Scenario [[…]]/>[[…]]

<Units><Unit [[…]]/>[[…]]

<Variables>
  <Variable id=[[…]]>
    <ForeignVariableReference [[…]]/>
  [[…]]
</Variables>

<CsvMappings><CsvMapping [[…]]>
  <TimeData [[…]]/>
  <ReferenceData variableRefId=[[…]]>
    <ColumnMapping
      unitRefId=[[…]] columnName=”u"/>
  [[…]]
```

```
time,u, [[…]]
0.0,0.0, [[…]]
0.001,1.5, [[…]]
```

*several possible

Behavioral Model features:

- Documentation & test scenarios in manifest
- Well-defined units and types
- Well-defined mapping of each variable’s dimensions to individual reference-trajectory (CSV column)
- Three types of tolerances, with well-defined interpretation:
  - absolute
  - relative
  - explicit upper and lower bound trajectories
- Two types of trajectories w.r.t. time:
  - equidistant (with well-defined restrictions on time trajectory tolerances)
  - variable (with well-defined interpolation)
- CSV reference trajectories strictly follow RFC 4180 (only “,” as separator, not “;”; only CRLF line-endings) with further restrictions:
  - no quoting, no additional whitespace
  - GALEC syntax for numbers
  - strictly monotone time trajectory

⇒ Unique interpretation

eFMI Standard: Container architecture & traceability

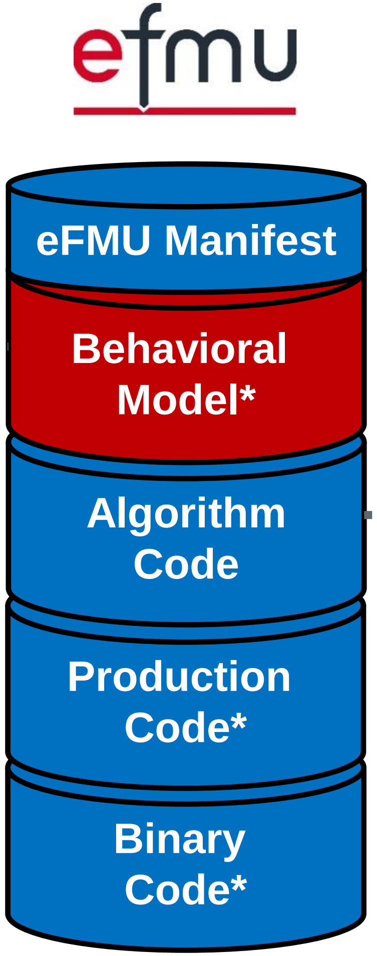
XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
  <ManifestReference
    id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Scenarios><Scenario [[…]]/>[[…]]

<Units><Unit [[…]]/>[[…]]

<Variables>
  <Variable id=[[…]]>
    <ForeignVariableReference [[…]]/>
  [[…]]
</Variables>

<CsvMappings><CsvMapping [[…]]>
  <TimeData [[…]]/>
  <ReferenceData variableRefId=[[…]]>
    <ColumnMapping
      unitRefId=[[…]] columnName="u"/>
  [[…]]
```

CSV

```
time,u, [[…]]
0.0,0.0, [[…]]
0.001,1.5, [[…]]
```

Trajectories
+
Meta information
(checksums, documentation, scenarios, tolerances, CSV & variable linkage)



*several possible

GALEC program

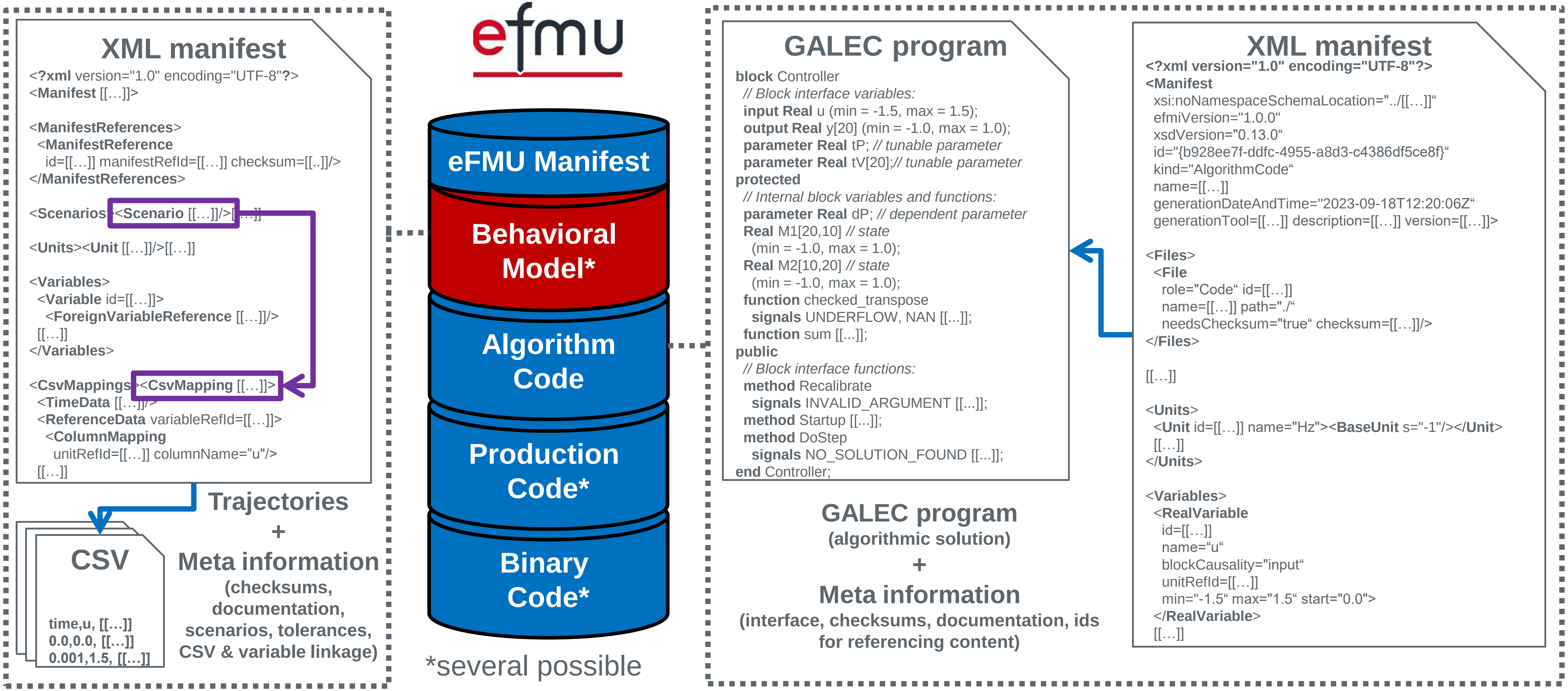
```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]"
  efmiVersion="1.0.0"
  xsdVersion="0.13.0"
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}"
  kind="AlgorithmCode"
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z"
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code" id=[[…]]
    name=[[…]] path="./"
    needsChecksum="true" checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name="u"
    blockCausality="input"
    unitRefId=[[…]]
    min="-1.5" max="1.5" start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability



*several possible

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
 <ManifestReference
   id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Scenarios><Scenario [[…]]/>[[…]]

<Units><Unit [[…]]/>[[…]]

<Variables>
 <Variable id=[[…]]>
   <ForeignVariableReference [[…]]/>
 [[…]]
</Variables>

<CsvMappings><CsvMapping [[…]]>
 <TimeData [[…]]/>
 <ReferenceData variableRefId=[[…]]>
   <ColumnMapping
     unitRefId=[[…]] columnName=”u"/>
 [[…]]
```

CSV

```
time,u, [[…]]
0.0,0.0, [[…]]
0.001,1.5, [[…]]
```

Trajectories
+
Meta information
(checksums, documentation, scenarios, tolerances, CSV & variable linkage)

GALEC program

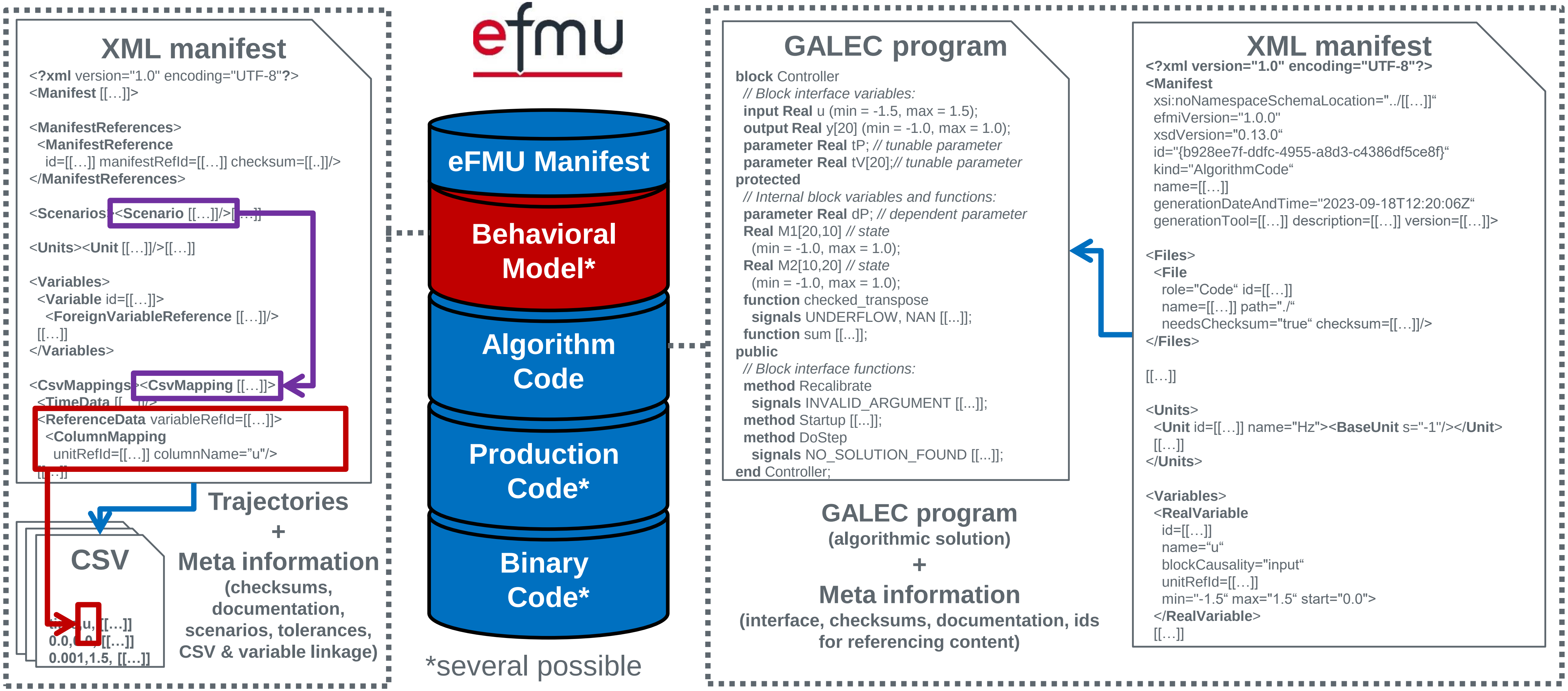
```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
 xsi:noNamespaceSchemaLocation="../[[…]]“
 efmiVersion="1.0.0"
 xsdVersion="0.13.0“
 id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
 kind="AlgorithmCode“
 name=[[…]]
 generationDateAndTime="2023-09-18T12:20:06Z“
 generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
 <File
   role="Code“ id=[[…]]
   name=[[…]] path="./“
   needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
 <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
 [[…]]
</Units>

<Variables>
 <RealVariable
   id=[[…]]
   name=“u“
   blockCausality="input“
   unitRefId=[[…]]
   min="-1.5“ max="1.5“ start="0.0">
 </RealVariable>
 [[…]]
```

eFMI Standard: Container architecture & traceability



*several possible

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
  <ManifestReference
   id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Scenarios><Scenario [[…]]/>[[…]]

<Units><Unit [[…]]/>[[…]]

<Variables>
  <Variable id=[[…]]>
    <ForeignVariableReference [[…]]/>
  [[…]]
</Variables>

<CsvMappings><CsvMapping [[…]]>
  <TimeData [[…]]/>
  <ReferenceData variableRefId=[[…]]>
   <ColumnMapping
    unitRefId=[[…]] columnName=”u"/>
  [[…]]
```

CSV

```
time,u, [[…]]
0.0,0.0, [[…]]
0.001,1.5, [[…]]
```

Trajectories

+

Meta information

(checksums, documentation, scenarios, tolerances, CSV & variable linkage)

GALEC program

```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program

(algorithmic solution)

+

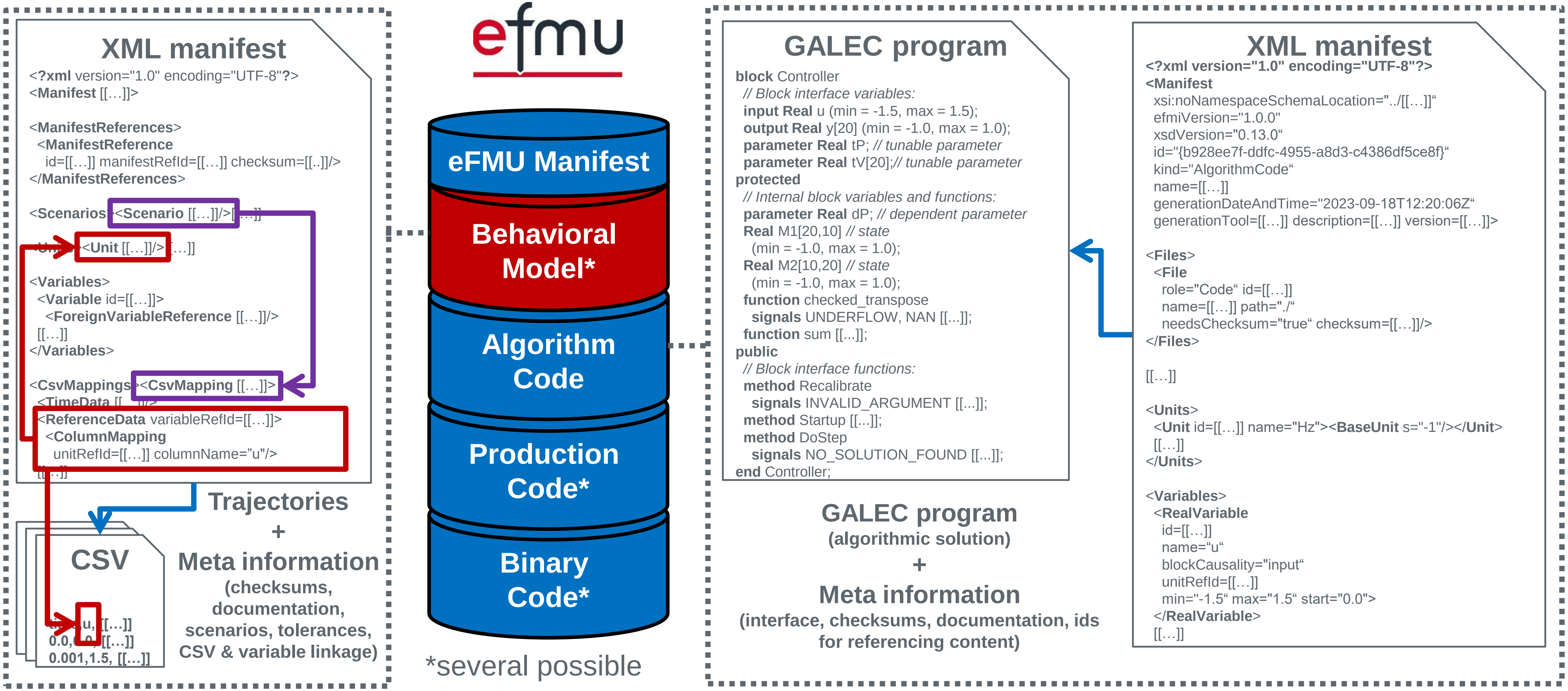
Meta information

(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability



XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
  <ManifestReference
    id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Scenarios><Scenario [[…]]/>[[…]]

<Units><Unit [[…]]/>[[…]]

<Variables>
  <Variable id=[[…]]>
    <ForeignVariableReference [[…]]/>
  [[…]]
</Variables>

<CsvMappings><CsvMapping [[…]]>
  <TimeData [[…]]/>
  <ReferenceData variableRefId=[[…]]>
    <ColumnMapping
      unitRefId=[[…]] columnName=”u"/>
  [[…]]
```

CSV

```
time,u, [[…]]
0.0,0.0, [[…]]
0.001,1.5, [[…]]
```

Trajectories
+
Meta information
(checksums, documentation, scenarios, tolerances, CSV & variable linkage)

*several possible

GALEC program

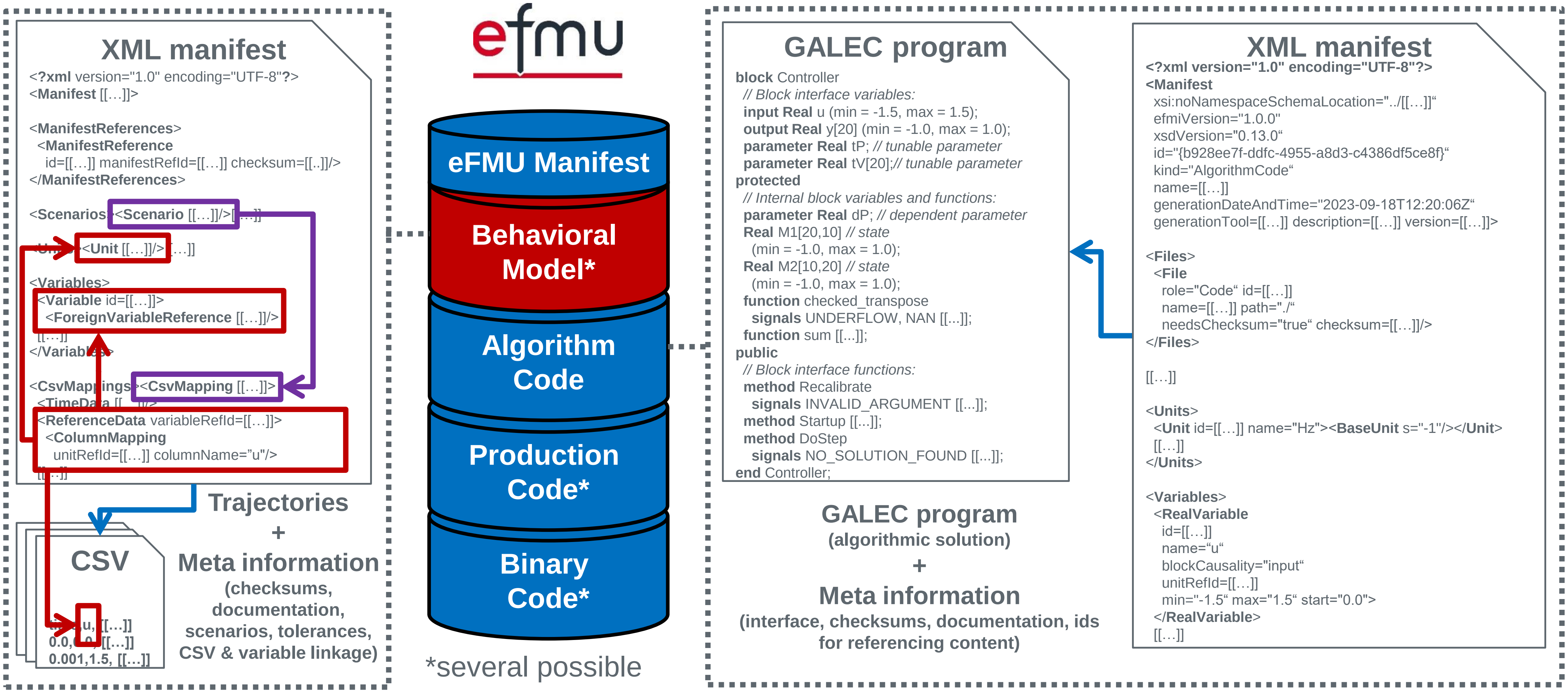
```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability



*several possible

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
 <ManifestReference
  id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Scenarios><Scenario [[…]]/>[[…]]

<Units><Unit [[…]]/>[[…]]

<Variables>
 <Variable id=[[…]]>
  <ForeignVariableReference [[…]]/>
 [[…]]
</Variables>

<CsvMappings><CsvMapping [[…]]>
 <TimeData [[…]]/>
 <ReferenceData variableRefId=[[…]]>
  <ColumnMapping
   unitRefId=[[…]] columnName="u"/>
 [[…]]
```

CSV

```
time,u, [[…]]
0.0,0.0, [[…]]
0.001,1.5, [[…]]
```

Trajectories
+
Meta information
(checksums, documentation, scenarios, tolerances, CSV & variable linkage)

GALEC program

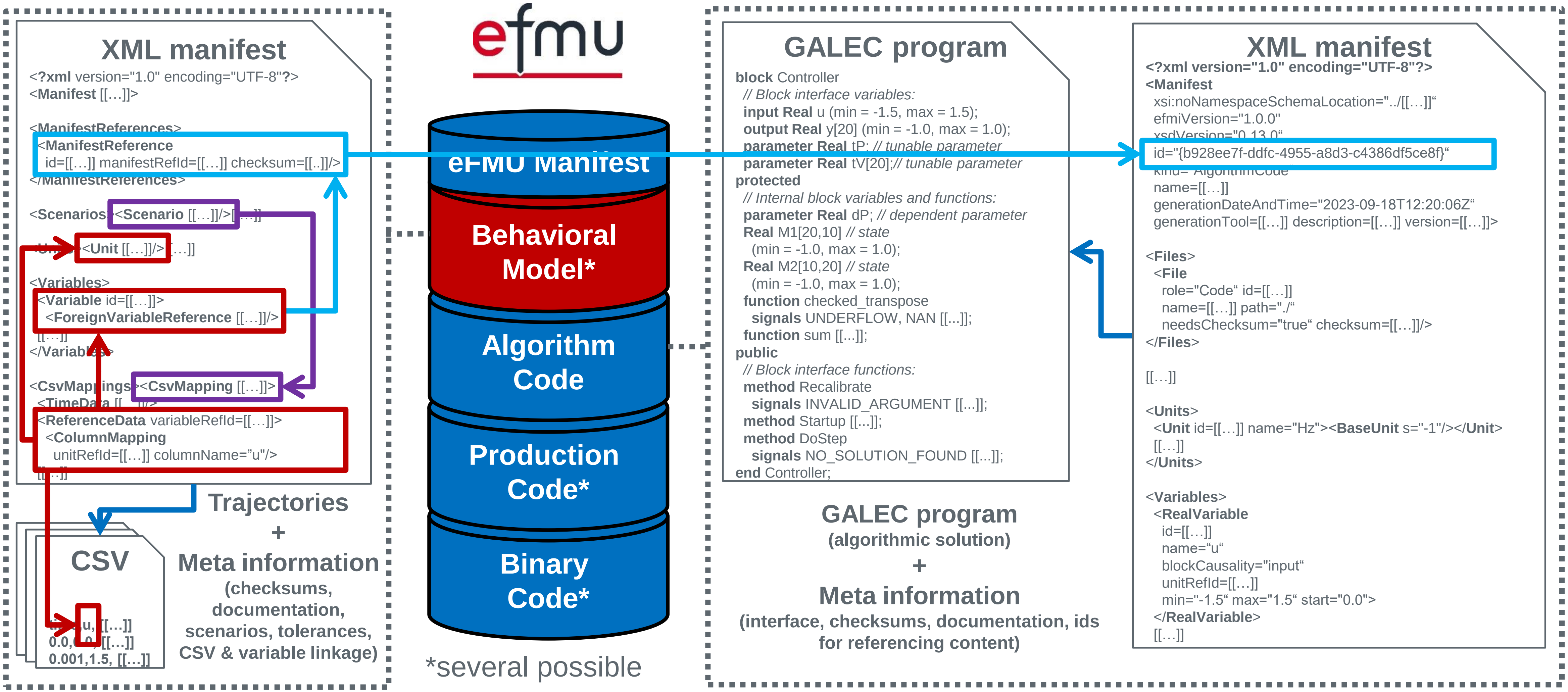
```
block Controller
 // Block interface variables:
 input Real u (min = -1.5, max = 1.5);
 output Real y[20] (min = -1.0, max = 1.0);
 parameter Real tP; // tunable parameter
 parameter Real tV[20];// tunable parameter
protected
 // Internal block variables and functions:
 parameter Real dP; // dependent parameter
 Real M1[20,10] // state
   (min = -1.0, max = 1.0);
 Real M2[10,20] // state
   (min = -1.0, max = 1.0);
 function checked_transpose
   signals UNDERFLOW, NAN [[...]];
 function sum [[...]];
public
 // Block interface functions:
 method Recalibrate
   signals INVALID_ARGUMENT [[...]];
 method Startup [[...]];
 method DoStep
   signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
 xsi:noNamespaceSchemaLocation="../[[…]]“
 efmiVersion="1.0.0"
 xsdVersion="0.13.0“
 id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
 kind="AlgorithmCode“
 name=[[…]]
 generationDateAndTime="2023-09-18T12:20:06Z“
 generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
 <File
  role="Code“ id=[[…]]
  name=[[…]] path="./“
  needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
 <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
 [[…]]
</Units>

<Variables>
 <RealVariable
  id=[[…]]
  name=“u“
  blockCausality="input“
  unitRefId=[[…]]
  min="-1.5“ max="1.5“ start="0.0">
 </RealVariable>
 [[…]]
```

eFMI Standard: Container architecture & traceability



XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
 <ManifestReference
  id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Scenarios><Scenario [[…]]/>[[…]]

<Units><Unit [[…]]/>[[…]]

<Variables>
 <Variable id=[[…]]>
  <ForeignVariableReference [[…]]/>
 [[…]]
</Variables>

<CsvMappings><CsvMapping [[…]]>
 <TimeData [[…]]/>
 <ReferenceData variableRefId=[[…]]>
  <ColumnMapping
   unitRefId=[[…]] columnName=”u"/>
 [[…]]
```

CSV

```
time,u, [[…]]
0.0,0.0, [[…]]
0.001,1.5, [[…]]
```

Trajectories
+
Meta information
(checksums, documentation, scenarios, tolerances, CSV & variable linkage)

*several possible

GALEC program

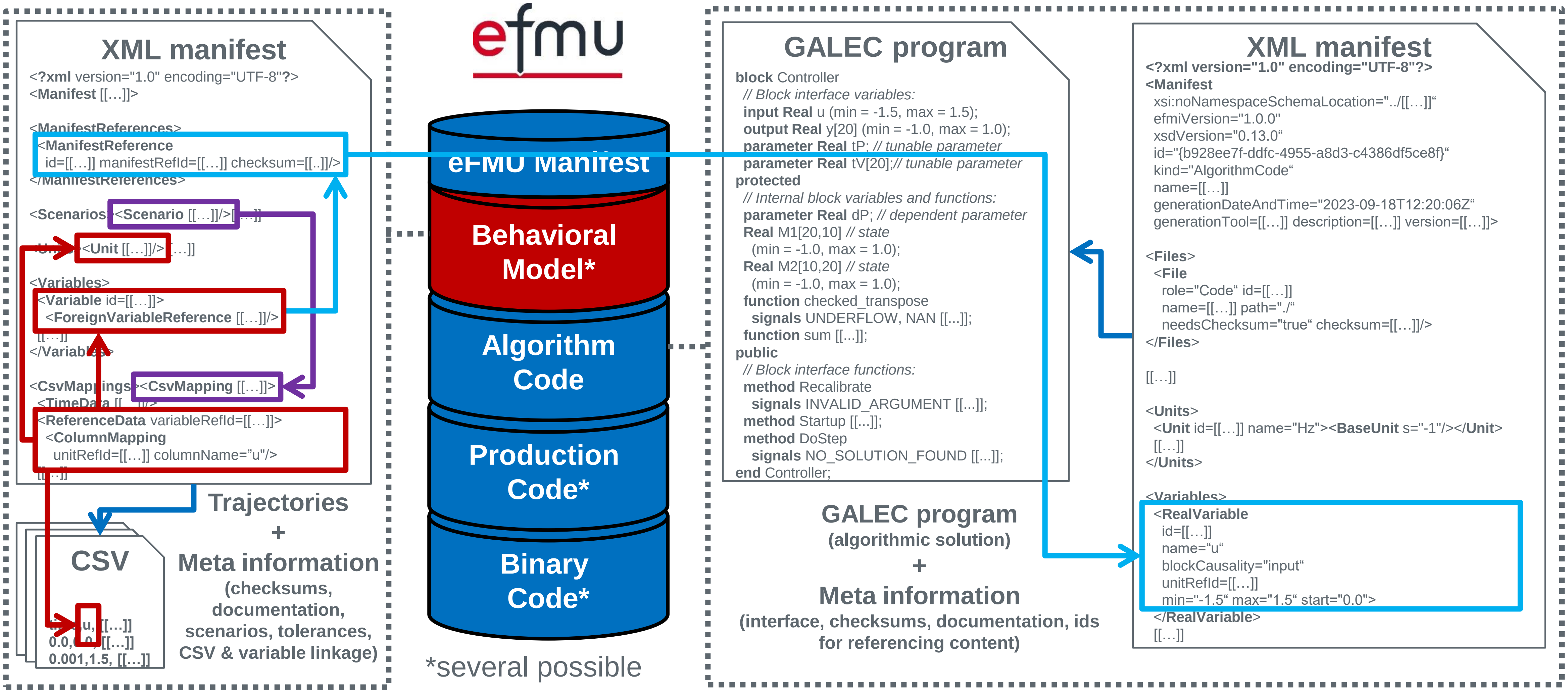
```
block Controller
 // Block interface variables:
 input Real u (min = -1.5, max = 1.5);
 output Real y[20] (min = -1.0, max = 1.0);
 parameter Real tP; // tunable parameter
 parameter Real tV[20];// tunable parameter
protected
 // Internal block variables and functions:
 parameter Real dP; // dependent parameter
 Real M1[20,10] // state
   (min = -1.0, max = 1.0);
 Real M2[10,20] // state
   (min = -1.0, max = 1.0);
 function checked_transpose
   signals UNDERFLOW, NAN [[...]];
 function sum [[...]];
public
 // Block interface functions:
 method Recalibrate
   signals INVALID_ARGUMENT [[...]];
 method Startup [[...]];
 method DoStep
   signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
 xsi:noNamespaceSchemaLocation="../[[…]]“
 efmiVersion="1.0.0"
 xsdVersion="0.13.0“
 id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
 kind="AlgorithmCode“
 name=[[…]]
 generationDateAndTime="2023-09-18T12:20:06Z“
 generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
 <File
  role="Code“ id=[[…]]
  name=[[…]] path="./“
  needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
 <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
 [[…]]
</Units>

<Variables>
 <RealVariable
  id=[[…]]
  name=“u“
  blockCausality="input“
  unitRefId=[[…]]
  min="-1.5“ max="1.5“ start="0.0">
 </RealVariable>
 [[…]]
```

eFMI Standard: Container architecture & traceability



XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
 <ManifestReference
  id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Scenarios><Scenario [[…]]/>[[…]]

<Units><Unit [[…]]/>[[…]]

<Variables>
 <Variable id=[[…]]>
  <ForeignVariableReference [[…]]/>
 [[…]]
</Variables>

<CsvMappings><CsvMapping [[…]]>
 <TimeData [[…]]/>
 <ReferenceData variableRefId=[[…]]>
  <ColumnMapping
   unitRefId=[[…]] columnName=”u"/>
 [[…]]
```

CSV

```
time,u, [[…]]
0.0,0.0, [[…]]
0.001,1.5, [[…]]
```

Trajectories
+
Meta information
(checksums, documentation, scenarios, tolerances, CSV & variable linkage)

*several possible

GALEC program

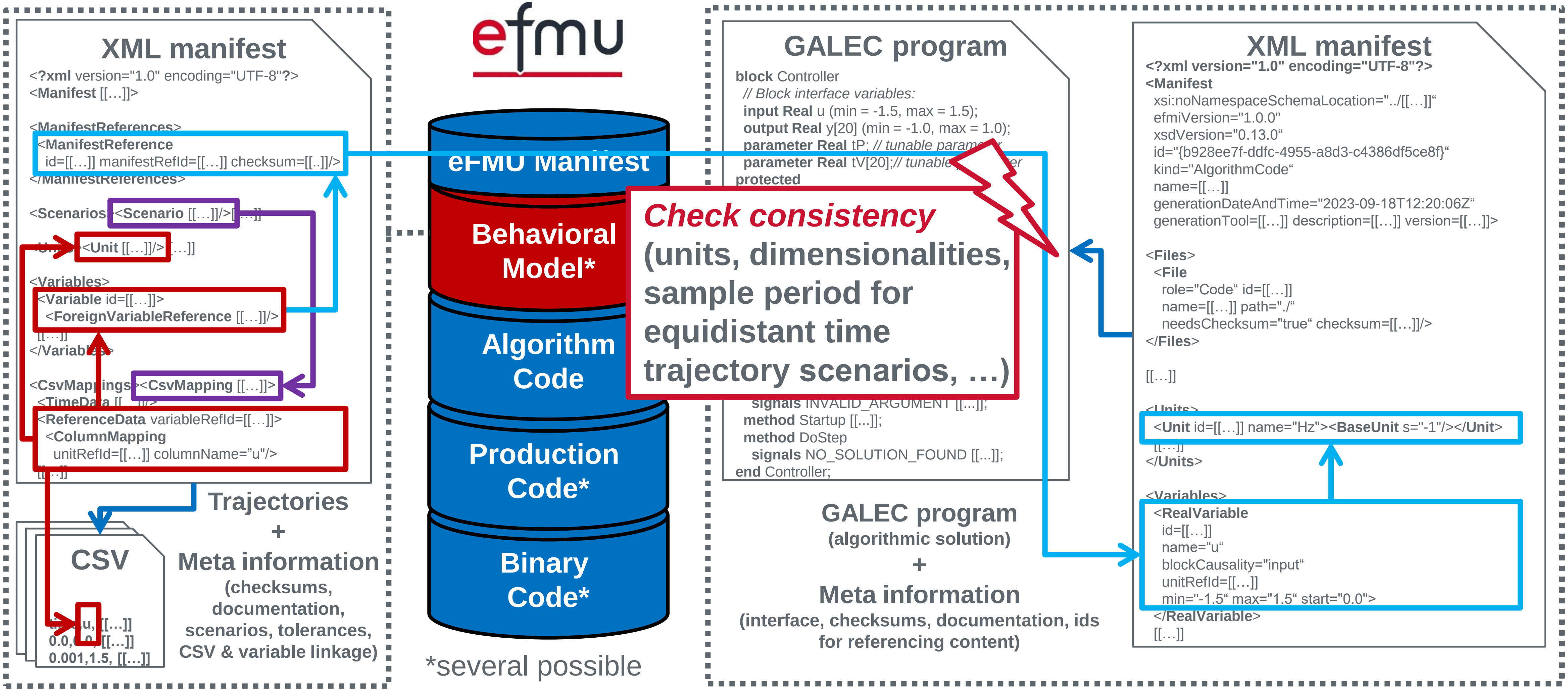
```
block Controller
 // Block interface variables:
 input Real u (min = -1.5, max = 1.5);
 output Real y[20] (min = -1.0, max = 1.0);
 parameter Real tP; // tunable parameter
 parameter Real tV[20];// tunable parameter
protected
 // Internal block variables and functions:
 parameter Real dP; // dependent parameter
 Real M1[20,10] // state
   (min = -1.0, max = 1.0);
 Real M2[10,20] // state
   (min = -1.0, max = 1.0);
 function checked_transpose
   signals UNDERFLOW, NAN [[...]];
 function sum [[...]];
public
 // Block interface functions:
 method Recalibrate
   signals INVALID_ARGUMENT [[...]];
 method Startup [[...]];
 method DoStep
   signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
 xsi:noNamespaceSchemaLocation="../[[…]]“
 efmiVersion="1.0.0"
 xsdVersion="0.13.0“
 id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
 kind="AlgorithmCode“
 name=[[…]]
 generationDateAndTime="2023-09-18T12:20:06Z“
 generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
 <File
  role="Code“ id=[[…]]
  name=[[…]] path="./“
  needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
 <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
 [[…]]
</Units>

<Variables>
 <RealVariable
  id=[[…]]
  name=“u“
  blockCausality="input“
  unitRefId=[[…]]
  min="-1.5“ max="1.5“ start="0.0">
 </RealVariable>
 [[…]]
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

eFMI Standard: Container architecture & traceability



```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
 <ManifestReference
  id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Scenarios><Scenario [[…]]/>[[…]]

<Units><Unit [[…]]/>[[…]]

<Variables>
 <Variable id=[[…]]>
  <ForeignVariableReference [[…]]/>
 [[…]]
</Variables>

<CsvMappings><CsvMapping [[…]]>
 <TimeData [[…]]/>
 <ReferenceData variableRefId=[[…]]>
  <ColumnMapping
   unitRefId=[[…]] columnName=”u"/>
 [[…]]
```

```
time,u, [[…]]
0.0,0.0, [[…]]
0.001,1.5, [[…]]
```

Trajectories
+
Meta information
(checksums, documentation, scenarios, tolerances, CSV & variable linkage)

*several possible

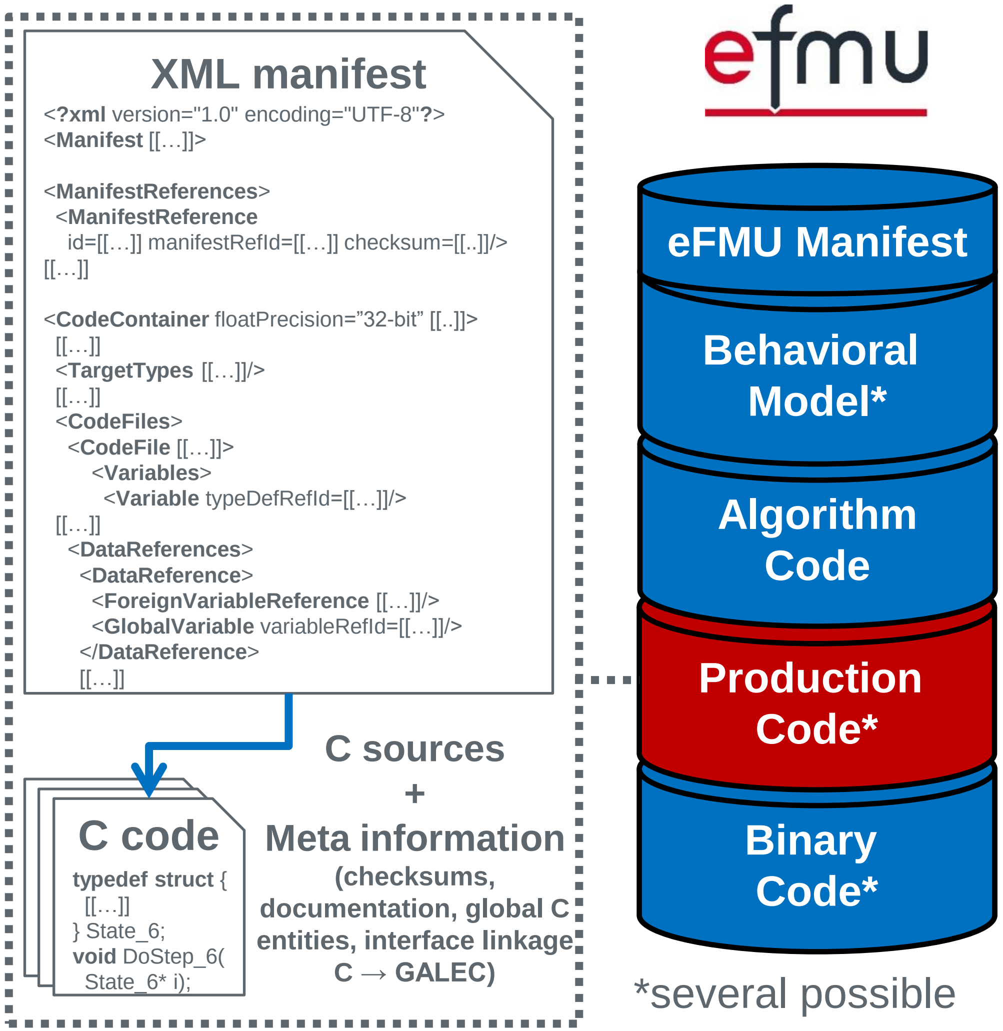
```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
[[…]]
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

Check consistency
(units, dimensionalities, sample period for equidistant time trajectory scenarios, …)

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability



XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
  <ManifestReference
    id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
[[…]]

<CodeContainer floatPrecision=”32-bit” [[..]]>
  [[…]]
  <TargetTypes [[…]]/>
  [[…]]
  <CodeFiles>
    <CodeFile [[…]]>
        <Variables>
          <Variable typeDefRefId=[[…]]/>
  [[…]]
    <DataReferences>
      <DataReference>
        <ForeignVariableReference [[…]]/>
        <GlobalVariable variableRefId=[[…]]/>
      </DataReference>
      [[…]]
```

C code

```
typedef struct {
  [[…]]
} State_6;
void DoStep_6(
  State_6* i);
```

*several possible

Production Code container features:

- Documentation of C sources, dependencies, global entities & interface in manifest
- C data layout & interface not standardized
  ⇒ Enables target environment tailoring
- Links C code to variables and functions of Algorithm Code manifest
  ⇒ Links implementation to GALEC block interface & life-cycle
  ⇒ Documents how to system integrate the production code

eFMI Standard: Container architecture & traceability

XML manifest

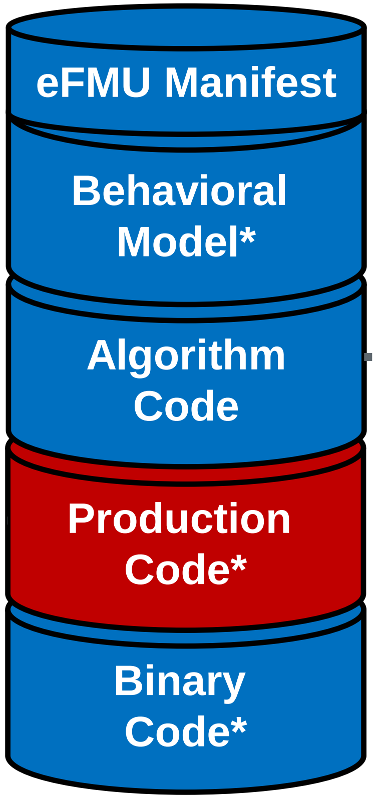
```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
  <ManifestReference
    id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
[[…]]

<CodeContainer floatPrecision=”32-bit” [[..]]>
  [[…]]
  <TargetTypes [[…]]/>
  [[…]]
  <CodeFiles>
    <CodeFile [[…]]>
        <Variables>
          <Variable typeDefRefId=[[…]]/>
  [[…]]
    <DataReferences>
      <DataReference>
        <ForeignVariableReference [[…]]/>
        <GlobalVariable variableRefId=[[…]]/>
      </DataReference>
      [[…]]
```

C code

```
typedef struct {
  [[…]]
} State_6;
void DoStep_6(
  State_6* i);
```

C sources
+
Meta information
(checksums, documentation, global C entities, interface linkage C → GALEC)

efmu



*several possible

GALEC program

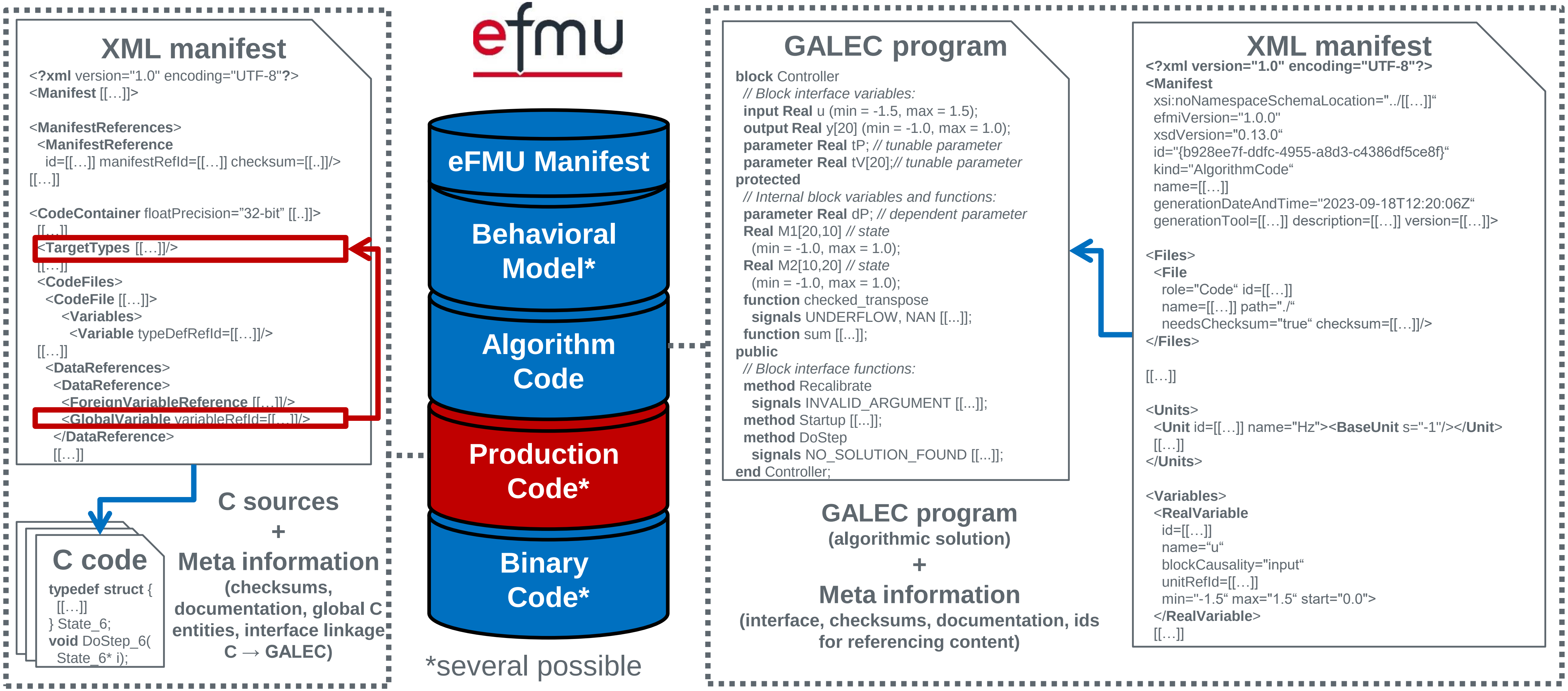
```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability



XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
  <ManifestReference
    id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
[[…]]

<CodeContainer floatPrecision=”32-bit” [[..]]>
  [[…]]
  <TargetTypes [[…]]/>
  [[…]]
  <CodeFiles>
    <CodeFile [[…]]>
        <Variables>
          <Variable typeDefRefId=[[…]]/>
  [[…]]
    <DataReferences>
      <DataReference>
        <ForeignVariableReference [[…]]/>
        <GlobalVariable variableRefId=[[…]]/>
      </DataReference>
      [[…]]
```

C code

```
typedef struct {
  [[…]]
} State_6;
void DoStep_6(
  State_6* i);
```

C sources
+
Meta information
(checksums, documentation, global C entities, interface linkage C → GALEC)

*several possible

GALEC program

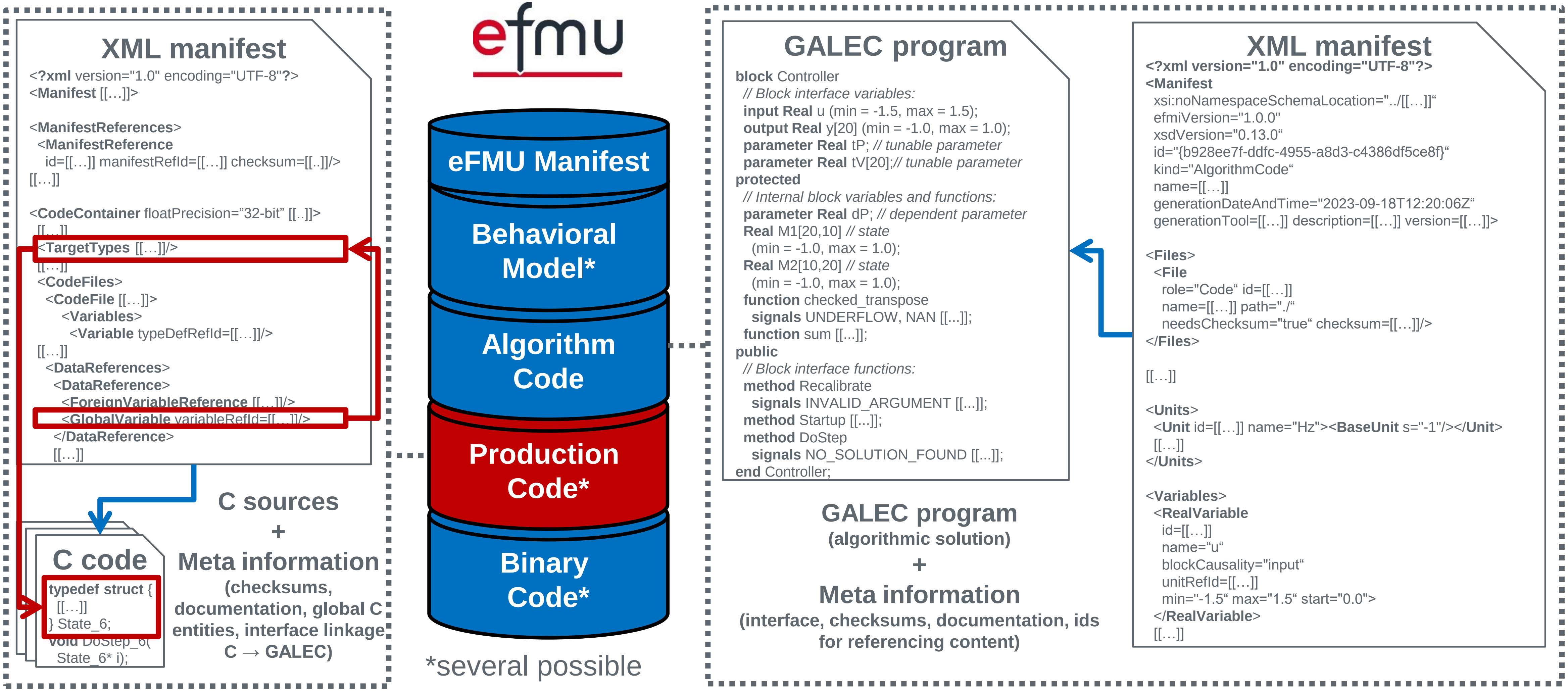
```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability



XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
  <ManifestReference
    id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
[[…]]

<CodeContainer floatPrecision=”32-bit” [[..]]>
  [[…]]
  <TargetTypes [[…]]/>
  [[…]]
  <CodeFiles>
    <CodeFile [[…]]>
        <Variables>
          <Variable typeDefRefId=[[…]]/>
  [[…]]
    <DataReferences>
      <DataReference>
        <ForeignVariableReference [[…]]/>
        <GlobalVariable variableRefId=[[…]]/>
      </DataReference>
      [[…]]
```

C code

```
typedef struct {
  [[…]]
} State_6;
void DoStep_6(
  State_6* i);
```

C sources
+
Meta information
(checksums, documentation, global C entities, interface linkage C → GALEC)

*several possible

GALEC program

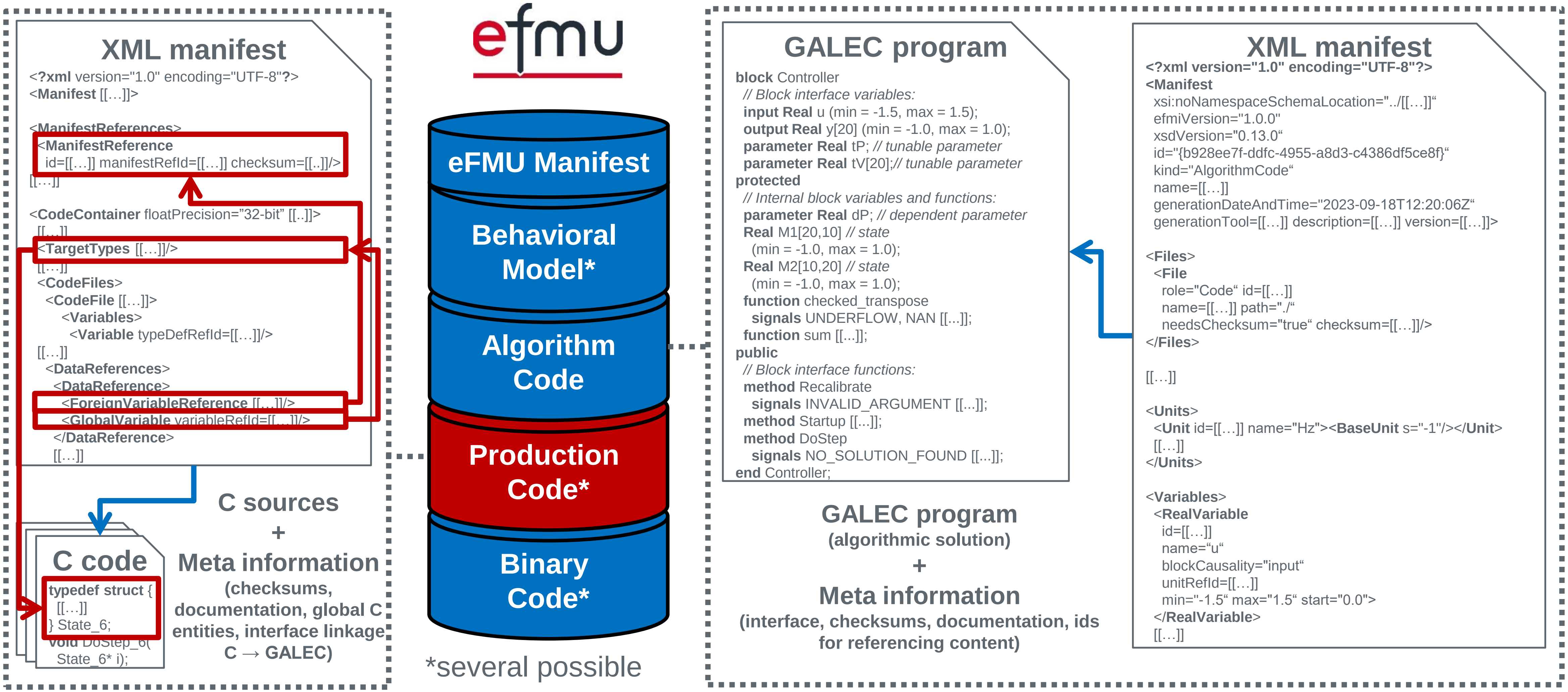
```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability



*several possible

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
 <ManifestReference
  id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
[[…]]

<CodeContainer floatPrecision=”32-bit” [[..]]>
 [[…]]
 <TargetTypes [[…]]/>
 [[…]]
 <CodeFiles>
  <CodeFile [[…]]>
    <Variables>
     <Variable typeDefRefId=[[…]]/>
 [[…]]
  <DataReferences>
   <DataReference>
    <ForeignVariableReference [[…]]/>
    <GlobalVariable variableRefId=[[…]]/>
   </DataReference>
   [[…]]
```

C code

```
typedef struct {
  [[…]]
} State_6;
void DoStep_6(
  State_6* i);
```

C sources
+
Meta information
(checksums, documentation, global C entities, interface linkage C → GALEC)

GALEC program

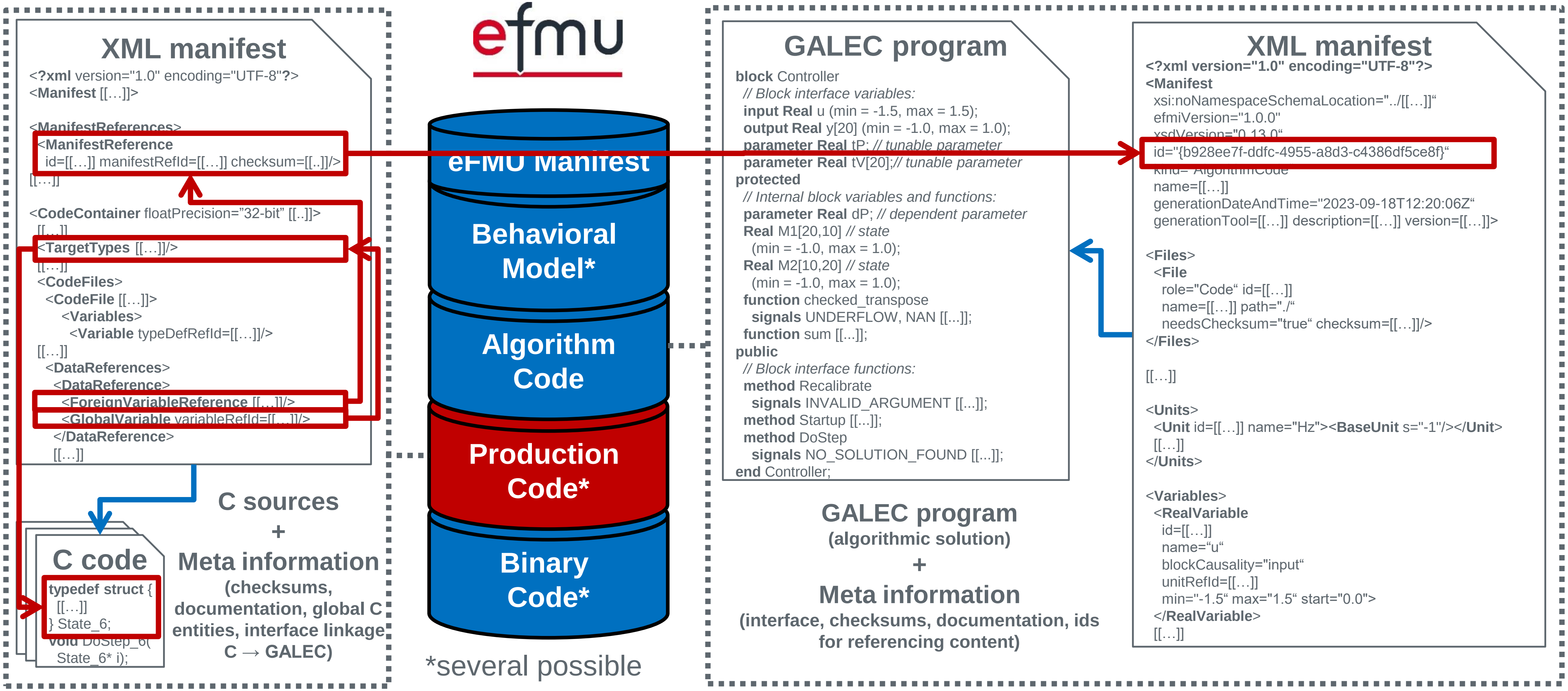
```
block Controller
  // Block interface variables:
  input Real u (min = -1.5, max = 1.5);
  output Real y[20] (min = -1.0, max = 1.0);
  parameter Real tP; // tunable parameter
  parameter Real tV[20];// tunable parameter
protected
  // Internal block variables and functions:
  parameter Real dP; // dependent parameter
  Real M1[20,10] // state
    (min = -1.0, max = 1.0);
  Real M2[10,20] // state
    (min = -1.0, max = 1.0);
  function checked_transpose
    signals UNDERFLOW, NAN [[...]];
  function sum [[...]];
public
  // Block interface functions:
  method Recalibrate
    signals INVALID_ARGUMENT [[...]];
  method Startup [[...]];
  method DoStep
    signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability



XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
 <ManifestReference
  id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
[[…]]

<CodeContainer floatPrecision=”32-bit” [[..]]>
 [[…]]
 <TargetTypes [[…]]/>
 [[…]]
 <CodeFiles>
  <CodeFile [[…]]>
    <Variables>
     <Variable typeDefRefId=[[…]]/>
 [[…]]
  <DataReferences>
   <DataReference>
    <ForeignVariableReference [[…]]/>
    <GlobalVariable variableRefId=[[…]]/>
   </DataReference>
   [[…]]
```

C code

```
typedef struct {
  [[…]]
} State_6;
void DoStep_6(
  State_6* i);
```

C sources
+
Meta information
(checksums, documentation, global C entities, interface linkage C → GALEC)

*several possible

GALEC program

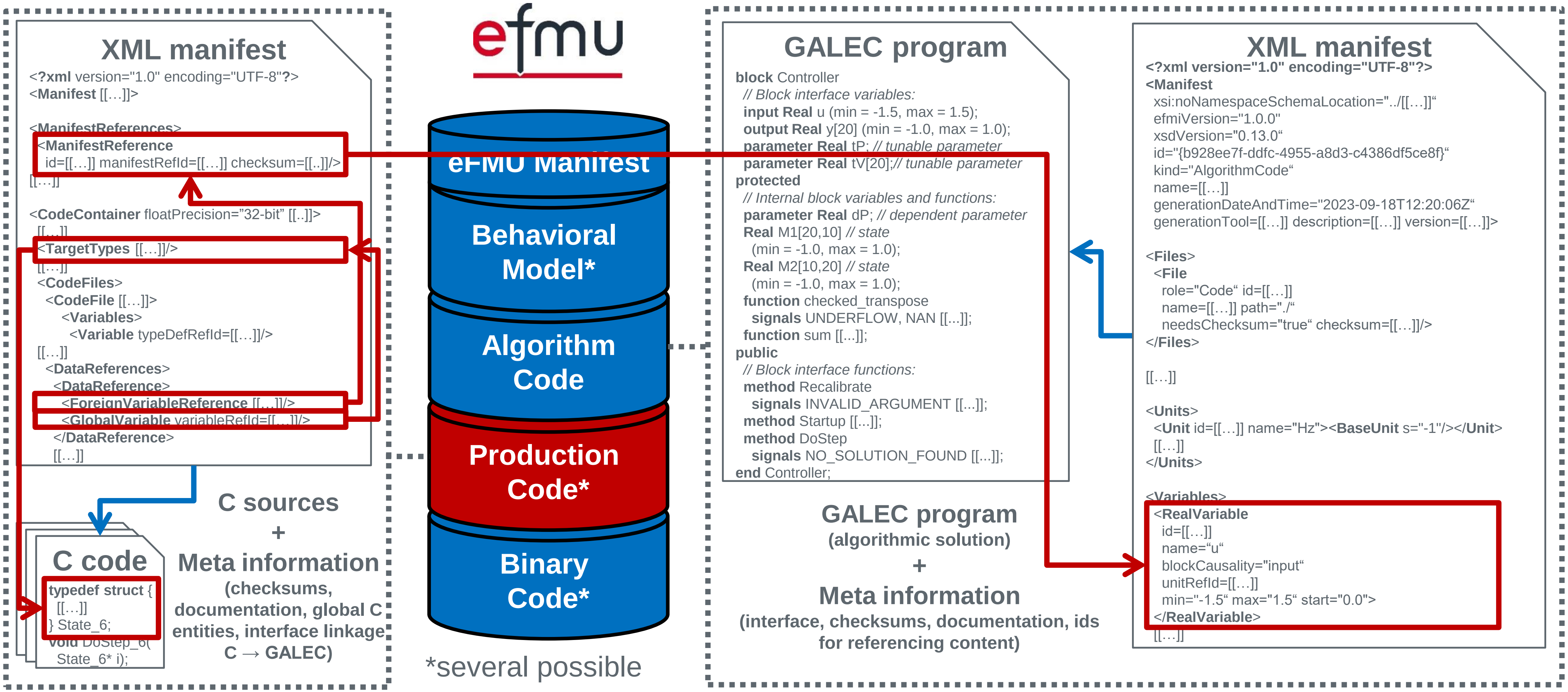
```
block Controller
 // Block interface variables:
 input Real u (min = -1.5, max = 1.5);
 output Real y[20] (min = -1.0, max = 1.0);
 parameter Real tP; // tunable parameter
 parameter Real tV[20];// tunable parameter
protected
 // Internal block variables and functions:
 parameter Real dP; // dependent parameter
 Real M1[20,10] // state
   (min = -1.0, max = 1.0);
 Real M2[10,20] // state
   (min = -1.0, max = 1.0);
 function checked_transpose
   signals UNDERFLOW, NAN [[...]];
 function sum [[...]];
public
 // Block interface functions:
 method Recalibrate
   signals INVALID_ARGUMENT [[...]];
 method Startup [[...]];
 method DoStep
   signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
  xsi:noNamespaceSchemaLocation="../[[…]]“
  efmiVersion="1.0.0"
  xsdVersion="0.13.0“
  id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
  kind="AlgorithmCode“
  name=[[…]]
  generationDateAndTime="2023-09-18T12:20:06Z“
  generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
  <File
    role="Code“ id=[[…]]
    name=[[…]] path="./“
    needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
  <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
  [[…]]
</Units>

<Variables>
  <RealVariable
    id=[[…]]
    name=“u“
    blockCausality="input“
    unitRefId=[[…]]
    min="-1.5“ max="1.5“ start="0.0">
  </RealVariable>
  [[…]]
```

eFMI Standard: Container architecture & traceability



efmu

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
 <ManifestReference
  id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
[[…]]

<CodeContainer floatPrecision=”32-bit” [[..]]>
 [[…]]
 <TargetTypes [[…]]/>
 [[…]]
 <CodeFiles>
  <CodeFile [[…]]>
    <Variables>
     <Variable typeDefRefId=[[…]]/>
 [[…]]
  <DataReferences>
   <DataReference>
    <ForeignVariableReference [[…]]/>
    <GlobalVariable variableRefId=[[…]]/>
   </DataReference>
   [[…]]
```

C code

```
typedef struct {
  [[…]]
} State_6;
void DoStep_6(
  State_6* i);
```

C sources
+
Meta information
(checksums, documentation, global C entities, interface linkage C → GALEC)

eFMU Manifest

Behavioral Model*

Algorithm Code

Production Code*

Binary Code*

*several possible

GALEC program

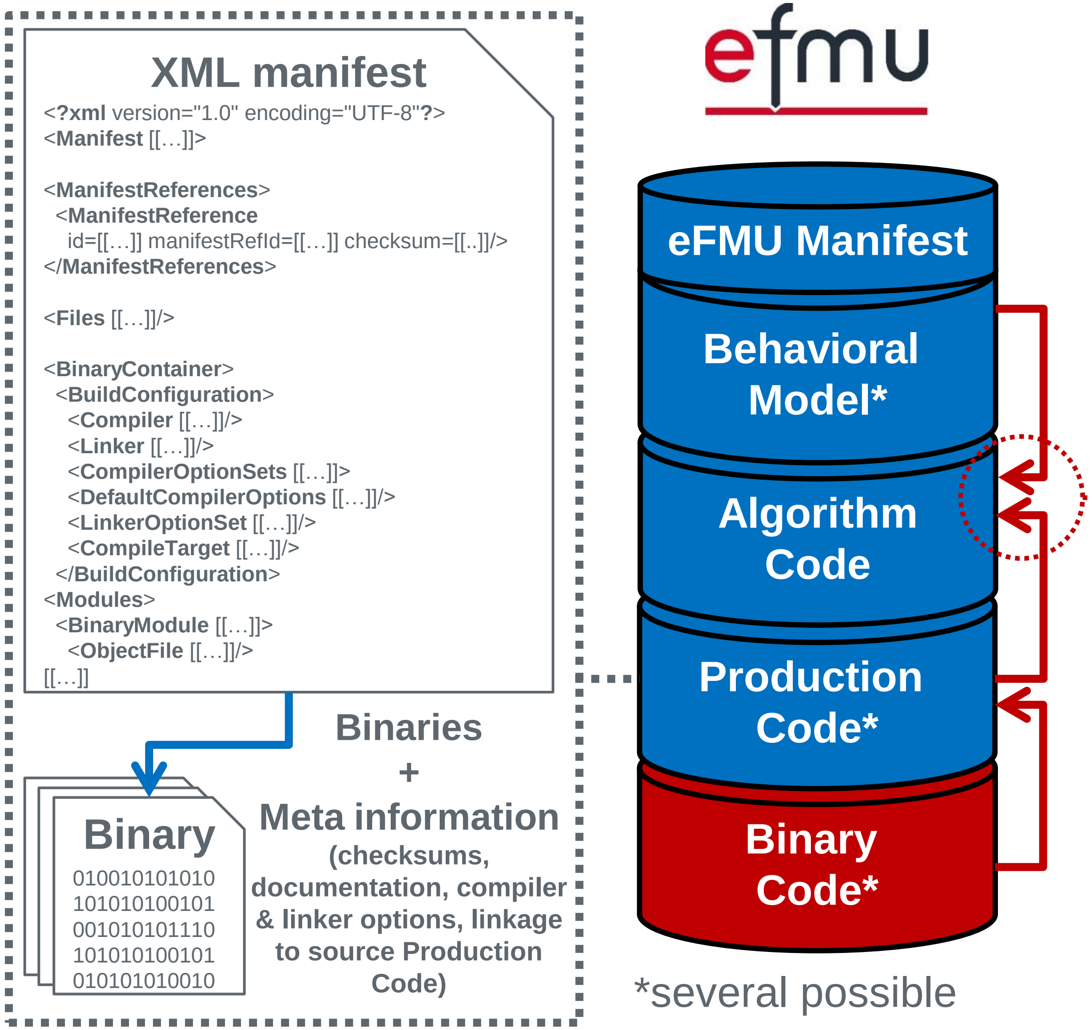
```
block Controller
 // Block interface variables:
 input Real u (min = -1.5, max = 1.5);
 output Real y[20] (min = -1.0, max = 1.0);
 parameter Real tP; // tunable parameter
 parameter Real tV[20];// tunable parameter
protected
 // Internal block variables and functions:
 parameter Real dP; // dependent parameter
 Real M1[20,10] // state
   (min = -1.0, max = 1.0);
 Real M2[10,20] // state
   (min = -1.0, max = 1.0);
 function checked_transpose
   signals UNDERFLOW, NAN [[...]];
 function sum [[...]];
public
 // Block interface functions:
 method Recalibrate
   signals INVALID_ARGUMENT [[...]];
 method Startup [[...]];
 method DoStep
   signals NO_SOLUTION_FOUND [[...]];
end Controller;
```

GALEC program
(algorithmic solution)
+
Meta information
(interface, checksums, documentation, ids for referencing content)

XML manifest

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
 xsi:noNamespaceSchemaLocation="../[[…]]“
 efmiVersion="1.0.0"
 xsdVersion="0.13.0“
 id="{b928ee7f-ddfc-4955-a8d3-c4386df5ce8f}“
 kind="AlgorithmCode“
 name=[[…]]
 generationDateAndTime="2023-09-18T12:20:06Z“
 generationTool=[[…]] description=[[…]] version=[[…]]>

<Files>
 <File
  role="Code“ id=[[…]]
  name=[[…]] path="./“
  needsChecksum="true“ checksum=[[…]]/>
</Files>

[[…]]

<Units>
 <Unit id=[[…]] name="Hz"><BaseUnit s="-1"/></Unit>
 [[…]]
</Units>

<Variables>
 <RealVariable
  id=[[…]]
  name=“u“
  blockCausality="input“
  unitRefId=[[…]]
  min="-1.5“ max="1.5“ start="0.0">
 </RealVariable>
 [[…]]
```

eFMI Standard: Container architecture & traceability



```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest [[…]]>

<ManifestReferences>
  <ManifestReference
    id=[[…]] manifestRefId=[[…]] checksum=[[..]]/>
</ManifestReferences>

<Files [[…]]/>

<BinaryContainer>
  <BuildConfiguration>
    <Compiler [[…]]/>
    <Linker [[…]]/>
    <CompilerOptionSets [[…]]>
    <DefaultCompilerOptions [[…]]/>
    <LinkerOptionSet [[…]]/>
    <CompileTarget [[…]]/>
  </BuildConfiguration>
<Modules>
  <BinaryModule [[…]]>
    <ObjectFile [[…]]/>
[[…]]
```

*several possible

Binary Code container features:

- Documentation of binaries & how they are build
  - used compiler & linker and their options
- Links to source Production Code container

To conduct tests:

- Match Behavioral Model with Production Code containers (via indirect linking)
- Use production code → GALEC interface linking for setup (i.e., for system integration in test environment)

eFMI Standard: Summary

The open standard for model-driven development of advanced control functions for safety-critical and real-time targets:

- Container architecture with well-defined model representations
  - Abstraction levels from expected behavior to binary code, from implementation to system-integration & testing

  ⇒ Enable collaboration of development stakeholders with different backgrounds, view-points & tooling (physics modeling, control engineering, embedded software development, etc)
  - Traceability & checksums

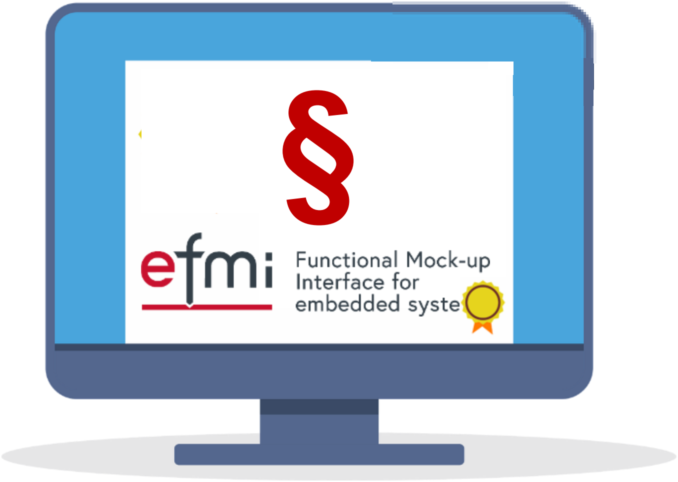
  ⇒ Enable detection of stale artefacts, toolchain automatization & code review
- GALEC with safety & real-time guarantees

  ⇒ Once algorithmic solution is found (not trivial modeling tool task), eFMI ”conveys” it to the embedded target (not trivial target environment tailoring & optimization task)
- Simple standard (only “what” has to be provided, not “how was it achieved”; no optional features)

  ⇒ The “Magic” is in the tools which are expert in their domain

efmi Functional Mock-up Interface for embedded systems

Congratulations, you got the basics of the eFMI Standard!



Now let’s move on to some practice.

eFMI® tutorial – Agenda

Part 1: eFMI® motivation and overview (40 min)

Part 2: Running use-case introduction (10 min)

Part 3: Hands-on in Dymola and Software Production Engineering (25 min)

Coffee break (30 min)

Part 3: Hands-on in Dymola and Software Production Engineering (30 min)

Part 4: Advanced demonstrators (20 min)

Part 5 (industry case-study): eFMI based thermal management system (TMS) development for fuel cell electric vehicles (FCEV) (20 min)

Part 6: Outlook and conclusion (5 min)

Tutorial leader:
Christoff Bürger

DASSAULT SYSTEMES

Presenter:
Daeoh Kang

iVH Institute of Vehicle Engineering

efmi Functional Mock-up Interface for embedded systems

License for

https://pixabay.com/illustrations/education-online-school-elearning-5307517/

I create these images with love and like to share them with you. My passion is to provide vintage designs to honor those artists that created something great and timeless. You are most welcome to use it for commercial projects, no need to ask for permission. I only ask that you not resell my images AS IS or claim them as your own creation. As always, a BIG thank you for the coffee donations I received, every dollar is a blessing for my family.

Education Online School royalty-free stock illustration. Free for use & download.

Content License Summary

Welcome to Pixabay! Pixabay is a vibrant community of authors, artists and creators sharing royalty-free images, video, audio and other media. We refer to this collectively as “**Content**”. By accessing and using Content, or by contributing Content, you agree to comply with our Content License.

At Pixabay, we like to keep things as simple as possible. For this reason, we have created this short summary of our Content License which is available in full here. Please keep in mind that only the full Content License is legally binding.

What are you allowed to do with Content?

- Subject to the Prohibited Uses (see below), the Content License allows users to:
- Use Content for free
- Use Content without having to attribute the author (although giving credit is always appreciated by our community!)
- Modify or adapt Content into new works

What are you not allowed to do with Content?

We refer to these as Prohibited Uses which include:

- You cannot sell or distribute Content (either in digital or physical form) on a Standalone basis. Standalone means where no creative effort has been applied to the Content and it remains in substantially the same form as it exists on our website.
- If Content contains any recognisable trademarks, logos or brands, you cannot use that Content for commercial purposes in relation to goods and services. In particular, you cannot print that Content on merchandise or other physical products for sale.
- You cannot use Content in any immoral or illegal way, especially Content which features recognisable people.
- You cannot use Content in a misleading or deceptive way.
- Please be aware that certain Content may be subject to additional intellectual property rights (such as copyrights, trademarks, design rights), moral rights, proprietary rights, property rights, privacy rights or similar. It is your responsibility to check whether you require the consent of a third party or a license to use Content.