CSD&M
5th International Conference
Complex Systems Design & Management 2014

MINES Alès

A verification approach from MDE applied to Model Based Systems Engineering: xeFFBD dynamic semantics

B. Nastov, V. Chapurlat, F. Pfister

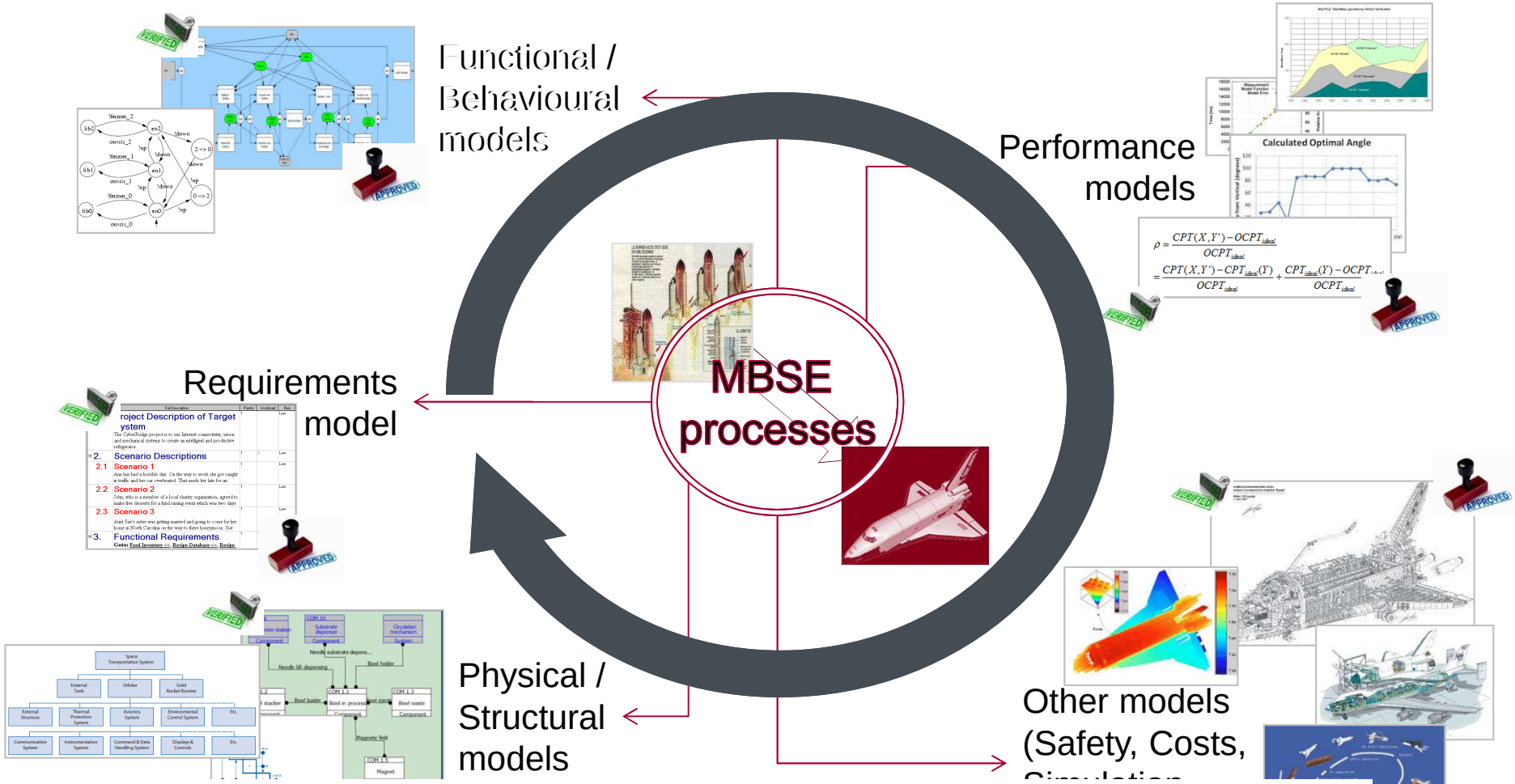
Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier
LIRMM

Christophe Dony

Context and problematic (1)

- **Systems Engineering (SE)** and **Model Based Systems Engineering (MBSE)**
- **Mainly, a model is…**
  - A partial image of (*a part of or a viewpoint of*) a system (existing or not)
  - Designed by an expert for reaching an objective: analyzing performance, choosing components, …
  - An instance of a Domain Specific Modelling Language (DSML) allowing to capture and to formalize information, knowledge and data considering the used viewpoint and for reaching various analysis objectives
  - ➢ Then, an arguing support for experts, in decisions making activities (architectural arrangement justification, respect to requirements, design constraints…)

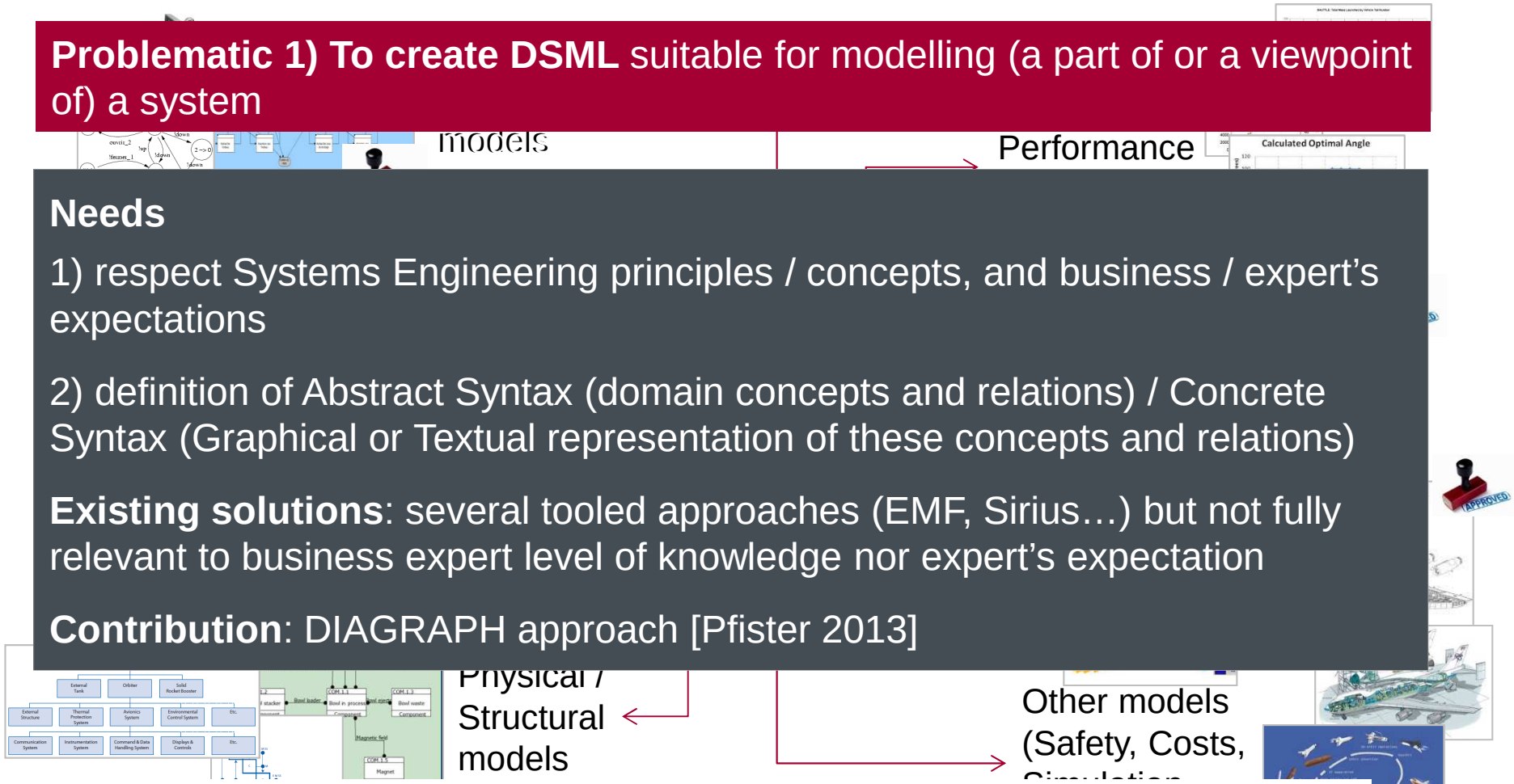
Context and problematic (2)



[Pfister 2013] F. Pfister, V. Chapurlat, H. Marianne, and C. Nebut, “A light-weight annotation-based solution to design Domain Specific Graphical Modeling Languages,” in Proceedings of 9th European Conference on Modelling Foundations and Applications ECMFA’’2013.

Context and problematic (2)

Problematic 1) To create DSML suitable for modelling (a part of or a viewpoint of) a system



Needs

1) respect Systems Engineering principles / concepts, and business / expert's expectations

2) definition of Abstract Syntax (domain concepts and relations) / Concrete Syntax (Graphical or Textual representation of these concepts and relations)

Existing solutions: several tooled approaches (EMF, Sirius…) but not fully relevant to business expert level of knowledge nor expert's expectation

Contribution: DIAGRAPH approach [Pfister 2013]

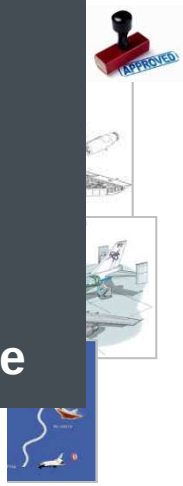
[Pfister 2013] F. Pfister, V. Chapurlat, H. Marianne, and C. Nebut, "A light-weight annotation-based solution to design Domain Specific Graphical Modeling Languages," in Proceedings of 9th European Conference on Modelling Foundations and Applications ECMFA''2013.

Context and problematic (2)

Problematic 2) To improve models acceptance: assume at least that these models are well formed, coherent, justifiable and (sufficiently) relevant for expert's objectives (verification / validation)

Needs

- Different aspects of a system are modeled by different models, so experts should take into account **simultaneously** all these models at least during system behavior simulation
- To facilitate expert's autonomy and confidence: to avoid M2M transformation, to be formal and automated or assisted as much as possible
- To take into account dynamics aspects of each DSML (operational semantics) and interdependencies between models
- To cover DSML **design time** and Model (instance of the DSML) **run time**



[Pfister 2013] F. Pfister, V. Chapurlat, H. Marianne, and C. Nebut, "A light-weight annotation-based solution to design Domain Specific Graphical Modeling Languages," in Proceedings of 9th European Conference on Modelling Foundations and Applications ECMFA''2013.

Context and problematic (3)

- **Existing solutions**
  - To guide and constraint modelling step (patterns, frameworks and pre defined models) or to expertise models: not formal, poorly assisted
  - To simulate, emulate or prove formally properties: not currently possible without M2M transformation, out of scope for SE experts
- **Contribution: to provide a conceptual and tooled approach allowing to formalize operational (dynamic) semantics of a DSML in order to facilitate automated models verification (and partial validation)**
  - **Simulation:** operational semantics, evolution algorithm and multi models synchronization mechanisms
  - **Proof**: properties modelling language and proof technique
  - For…
    - **DSML Design Time**: proving coherence and relevance of operational semantics
    - **Model Run time**: expected System's properties and global behaviour

Contents

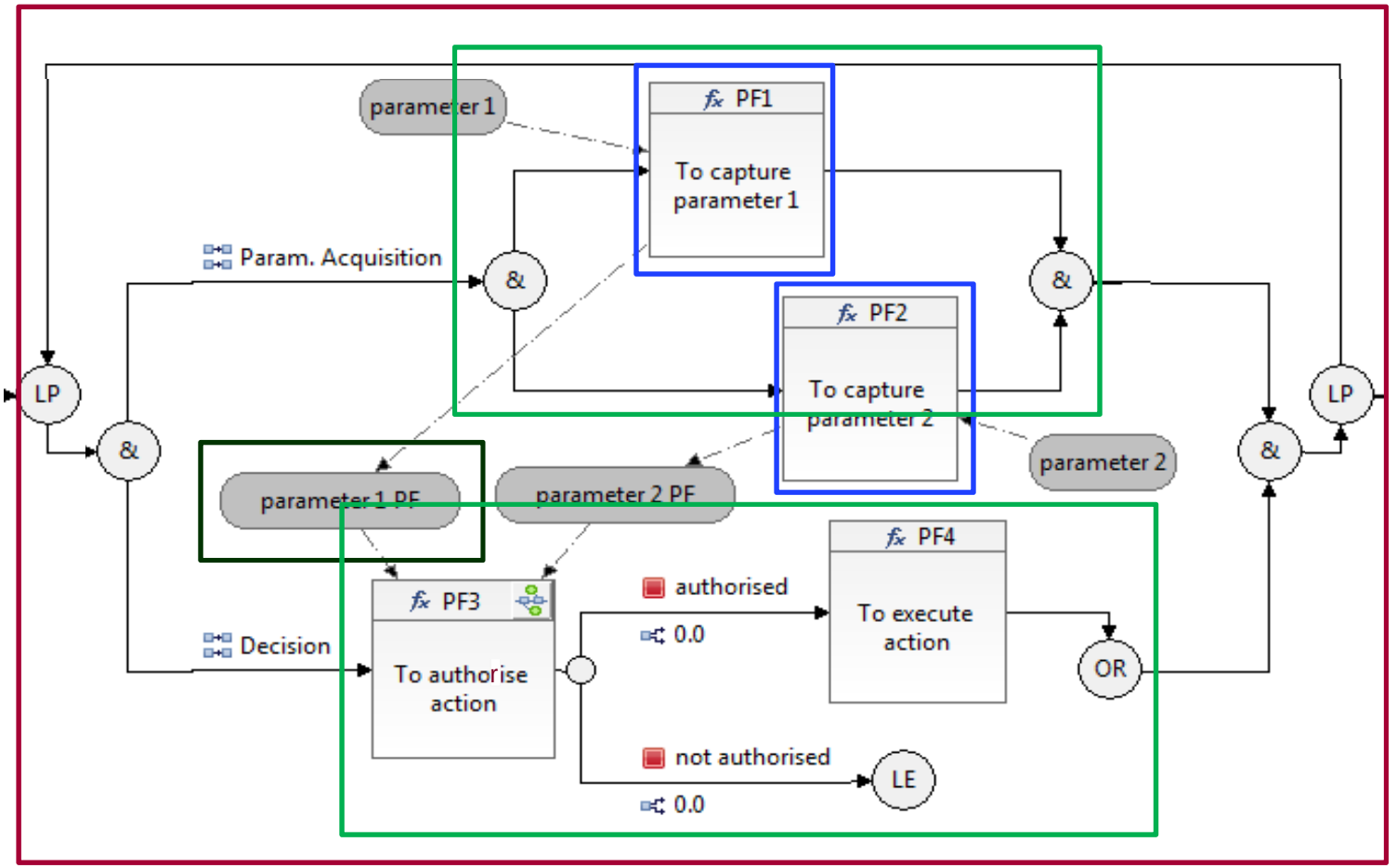
- **Goal:** to test, discuss and propose enrichments to be carried out on an existing “executable DSML” building approach
- **Approach applied:**
  - Illustrative example of DSML: enhanced Functional Flows Block Diagram (eFFBD) modelling language
  - From eFFBD to xeFFBD
- **Discussion**
  - Results experimentation and detected locks / limitations
  - Proposed contributions
  - Example of contributions

enhanced Functional Flows Block Diagram (eFFBD) modelling language

Function

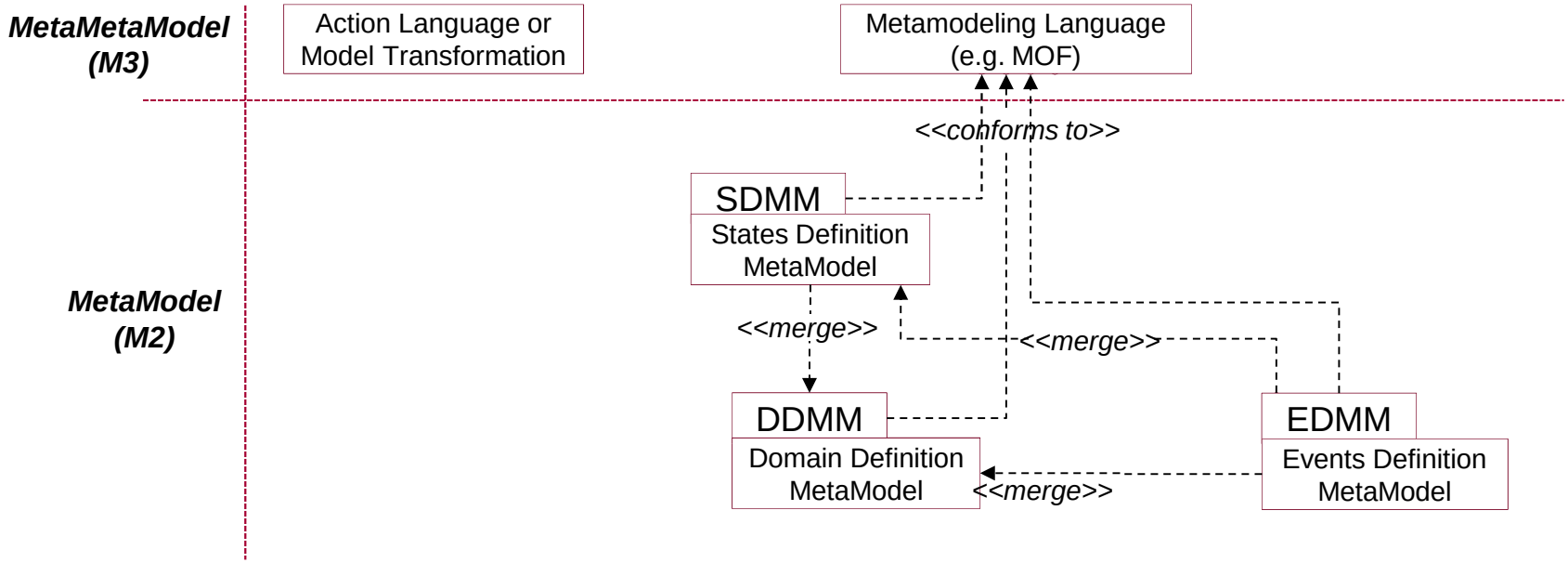
Flow
(I/O, Trigger)

Construct
(parallelism, choice, loop, …)



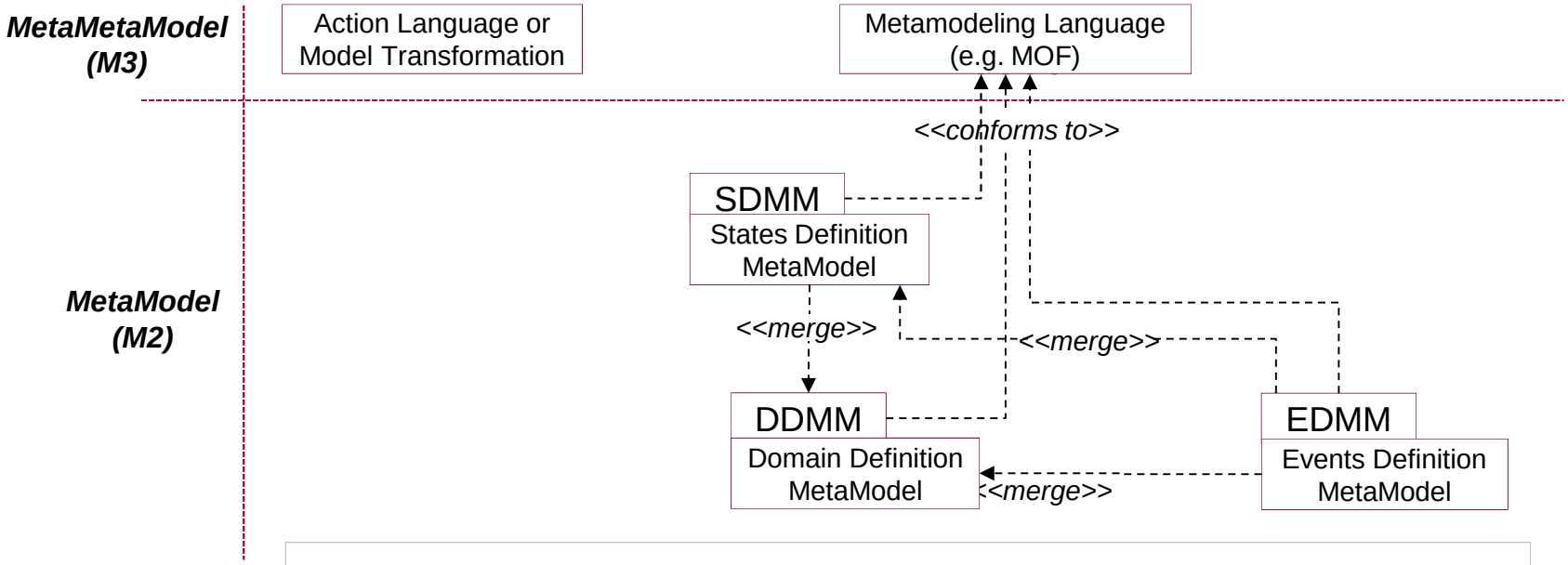
Functional / Behavioural model (dynamics of the system)

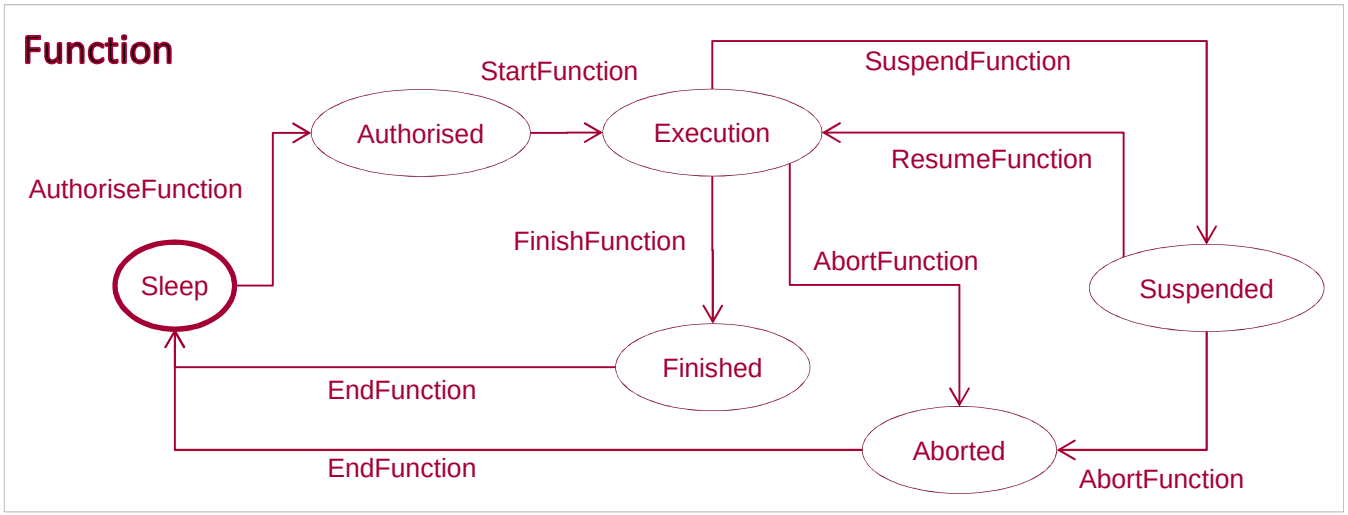
Basic approach



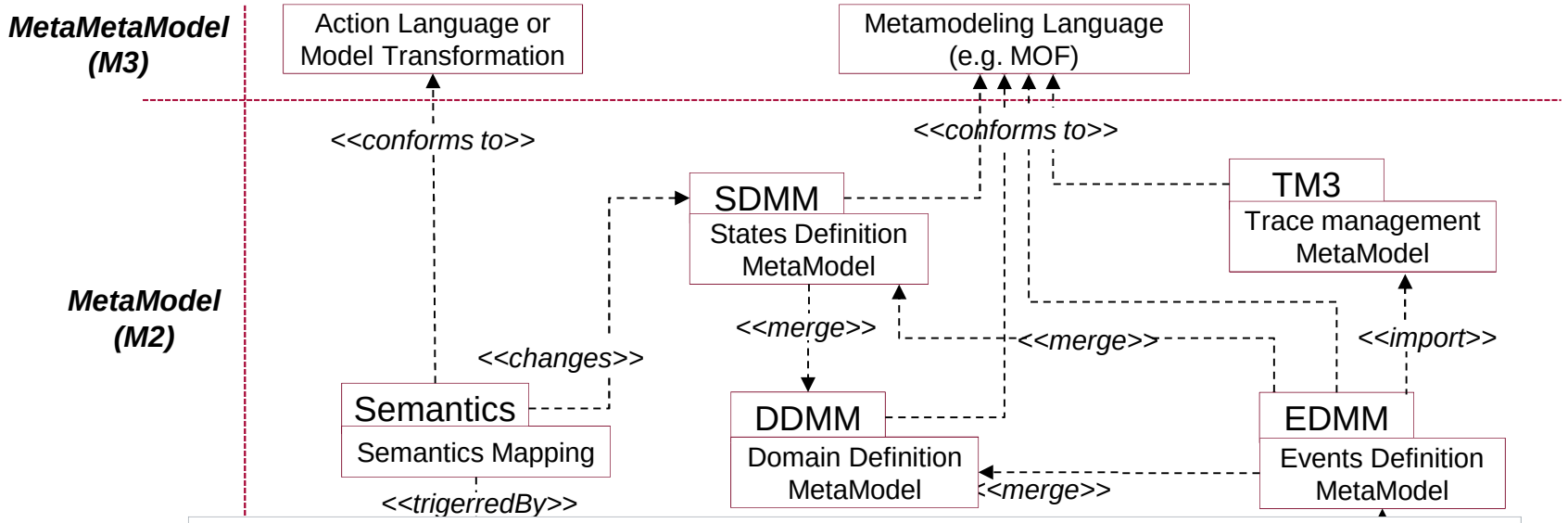
[Combemale 2012] A Design Pattern to Build Executable DSMLs and Associated V&V Tools
[Combemale 2008] A Property-Driven Approach for Formal Verification of process Models

Basic approach





Basic approach

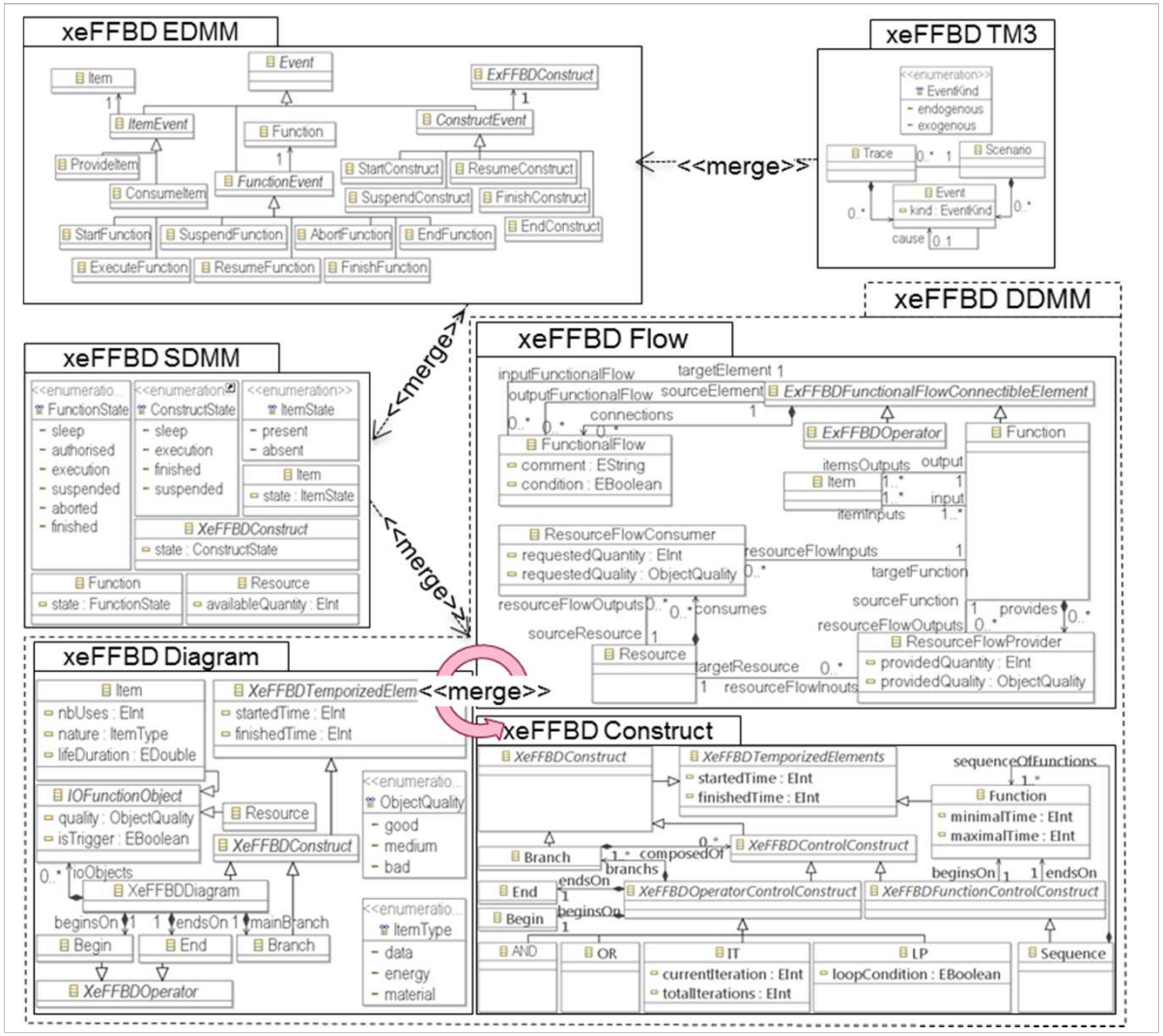


For f ∈ Function

(Eq. 1)	{ (f.state==***authorised***) **AND** (∀ i ∈ f.itemInputs,(i.state==***present***)) **AND** (∀ j ∈ f. resourceFlowInputs, ((j.requestedQuantity >= j.sourceResource.availableQuantity) **AND** (j.requestedQuality == j.sourceResource.quality)))) **implies** ***executeFunction***(f) }
(Eq. 2)	{ (f.state==***execution***) **implies** ((∀ i ∈ f. itemInputs, (***consumeItem***(i))) **AND** (∀ j ∈ f. resourceFlowInputs, (j.sourceResource.availableQuantity **-=** j.requestedQuantity)) }
(Eq. 3)	{ ((f.state==***execution***) **AND** ((internalTime - f.startedTime) >= minimalTime) **AND** ((internalTime - f.startedTime) <= maximalTime)) **implies** (***finishFunction***(f)) }
(Eq. 4)	{ (f.state==***finished***) **implies** ((∀ i ∈ f. itemOutputs, (***provideItem***(i))) **AND** (∀ j ∈ f. resourceFlowOutputs, (j.targetResource.availableQuantity **+=** j.providedQuantity)))}

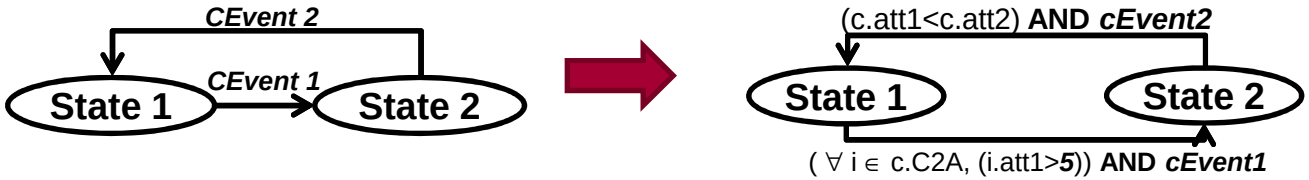
MINES Alès

Basic approach



Discussion and Perspectives

- **Experiment results:** enrichments and work under development
  - **Separating control part / data part** (reducing state machine complexity, improving representation power, formalisation level)
    - **Transition firing**: event based -> condition and event based



Discussion and Perspectives

- **Experiment results:** enrichments and work under development
  - **Separating control part / data part** (reducing state machine complexity, improving representation power, formalisation level)
    - **Transition firing**: event based -> condition and event based
    - **State notion**: discrete -> hybrid representation (continuous / discrete)

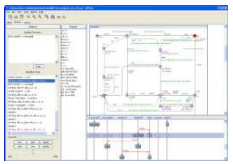
Function (partial)

Resource

availabilityLevel [0.. maxAvailabilityLevel]
qualityLevel [0..maxQualtityLevel]
maxAvailabilityLevel : Real: default = 10.0;
maxQualtityLevel : Real: default = 10.0;
computeLevel(requestFunctions, currentExternalTimeRequest)
isAvailable(minLevel)
isOffQuality(minLevel)
…

Discussion and Perspectives

- **Experiment results:** enrichments and work under development
  - **Separating control part / data part** (reducing state machine complexity, improving representation power, formalisation level)
    - **Transition firing**: event based -> condition and event based
    - **State notion**: discrete -> hybrid representation (continuous / discrete)
  - **Tooling** (for experimentations)




eFFBD: enhanced Functional Flows Block Diagram - **PBD**: Physical Block Diagram - **PN**: Petri Nets - **TPN**: Temporised Petri Nets - **FSM**: Finite State Machine - **LD**: Ladder Diagram – **GEMOS**: Guide d'Etude des Modes Opérationnels des Systèmes

Discussion and Perspectives

- **Experiment results:** enrichments and work under development
  - **Separating control part / data part** (reducing state machine complexity, improving representation power, formalisation level)
    - **Transition firing**: event based -> condition and event based
    - **State notion**: discrete -> hybrid representation (continuous / discrete)
  - **Tooling** (for experimentations)
  - **Model stability**: how detecting transient states? temporal hypothesis and simulation algorithm
  - **Proof**
    - Properties modelling language dedicated to DSML
    - Properties checking Technique / Parallel simulation technique

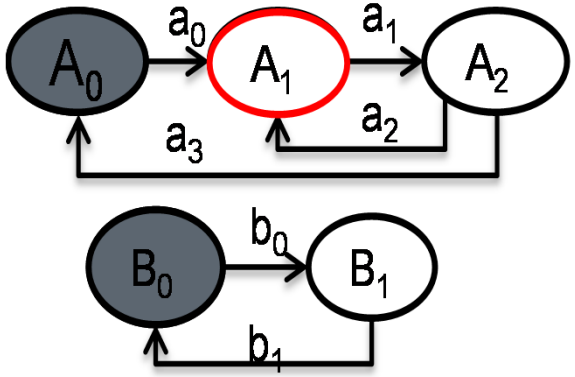
Transient State and model stability

- **Transient state S (contrary stable):** there exists an output transition of S that can be fired without any modification of input of the model M containing S
- **Stable model M:** the current state S of M is stable

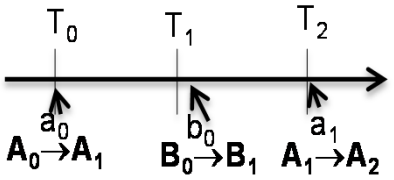
Firing Conditions

- $b_0 = \uparrow A_1$
- $a_1 = \uparrow B_1$

Scenario = (a_0, T_0, A_0, B_0)

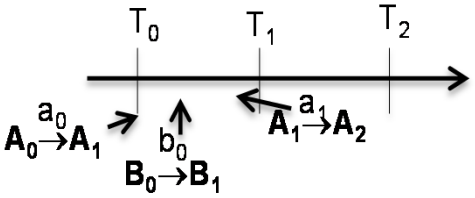


Without stability management



Result (T_3, A_2, B_1)

With stability management



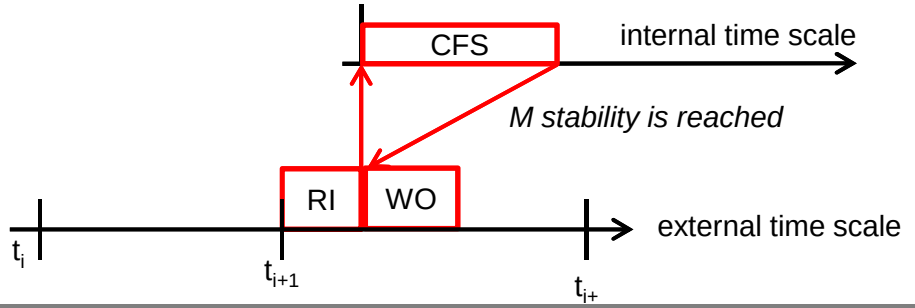
Result (T_1, A_2, B_1)

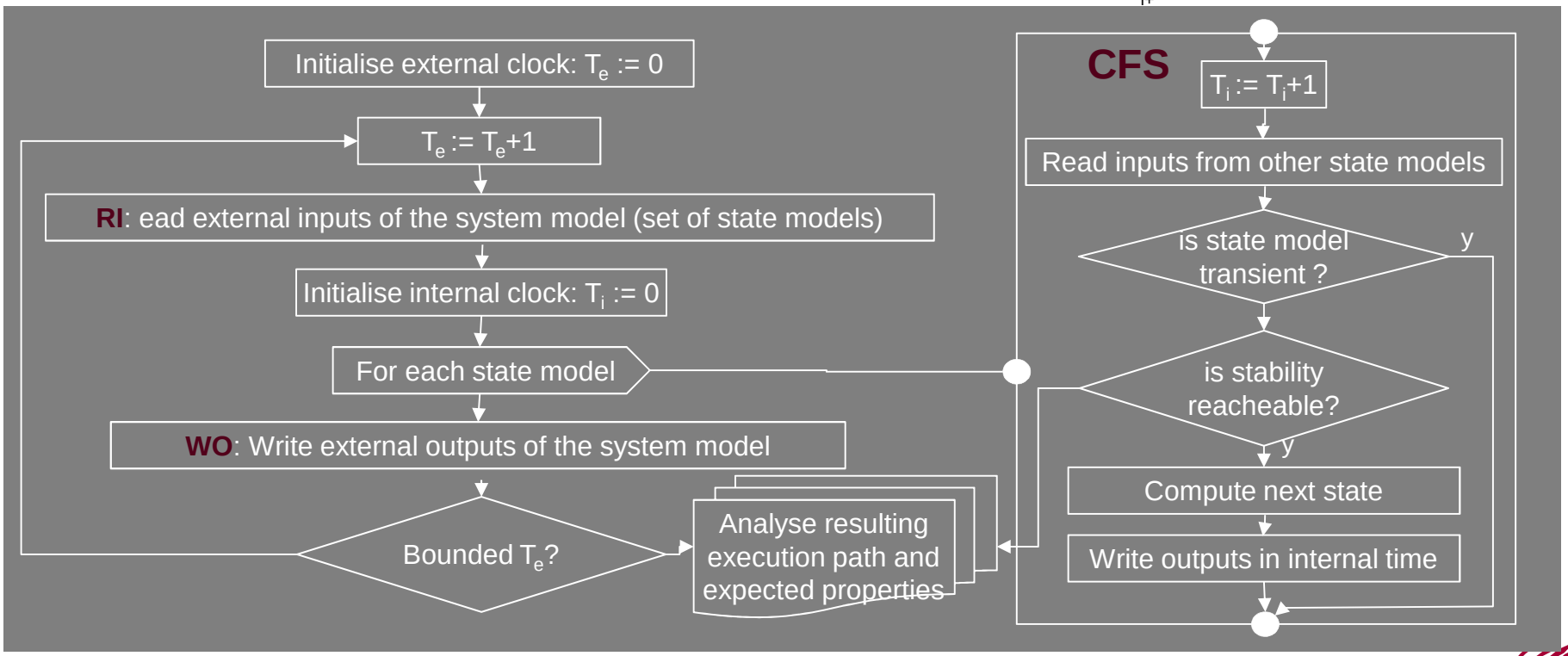
Transient State and model stability

- **Transient state S (contrary stable):** there exists an output transition of S that can be fired without any modification of input of the model M containing S
- **Stable model M:** the current state S of M is stable
- **Enrichment:** introducing « internal » and « external » time scales
  - **External**: depends from physical time scale (system's environment time scale)
  - **Internal**: fully independent from external and defined to reach model stability or detect transient states
  - Proposal of an **evolution algorithm with stability search** for simulating synchronization of behaviour of various concepts from
    - Same DMSL: internal synchronization
    - Various DMSLs: model synchronization

Simulation algorithm

RI – reading inputs of M
CFS – compute future state
WO – write outputs





MINES Alès

Many thanks