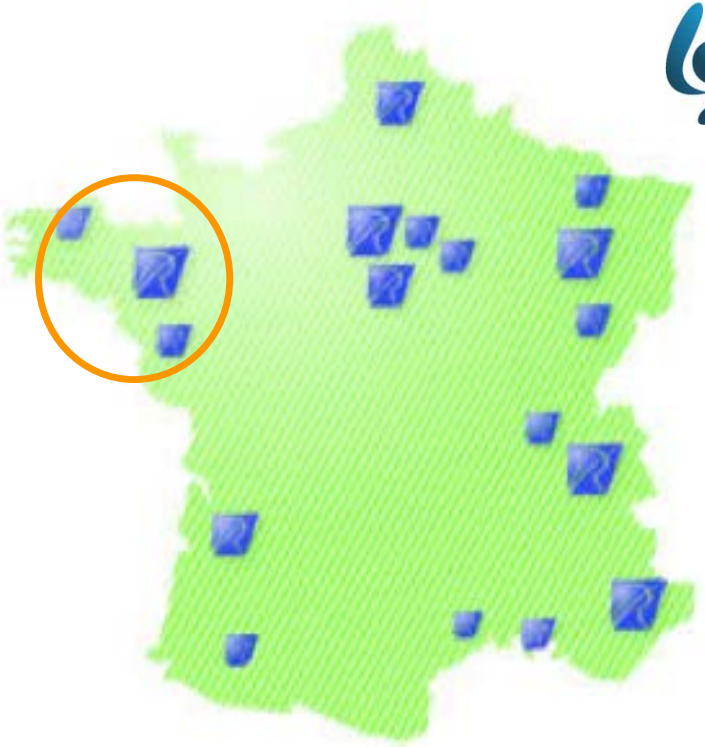
Composition and concurrent execution of heterogeneous domain-specific models

A work part of the GEMOC initiative

Benoit Combemale
Associate Professor, University of Rennes 1
Research Scientist, INRIA
benoit.combemale@irisa.fr

Journée IDM et modèles scientifiques, 18 octobre 2013

TRISKELL - team



UNIVERSITÉ DE RENNES 1

informatics mathematics

INSA RENNES

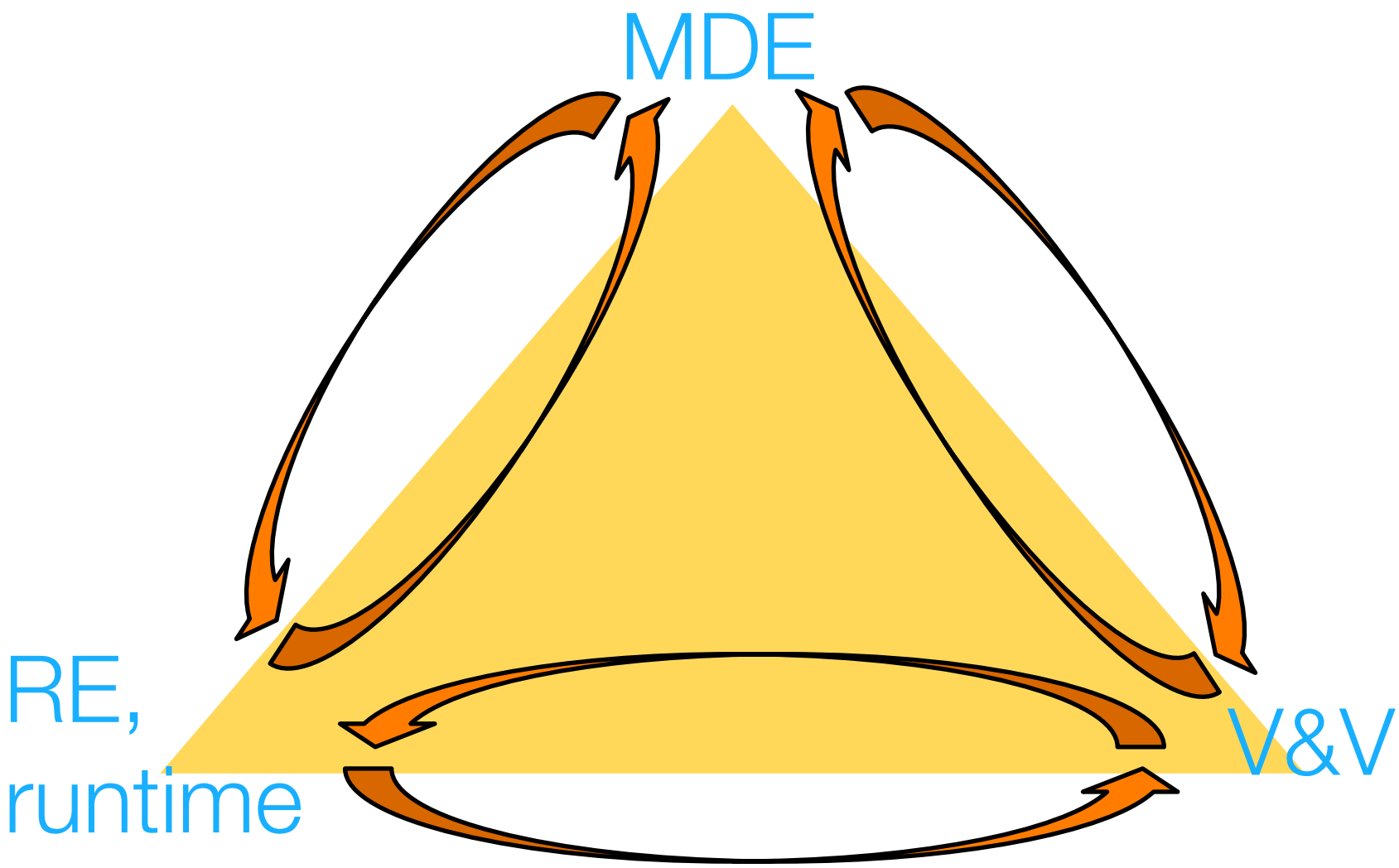
UMR IRISA

cnrs dépasser les frontières

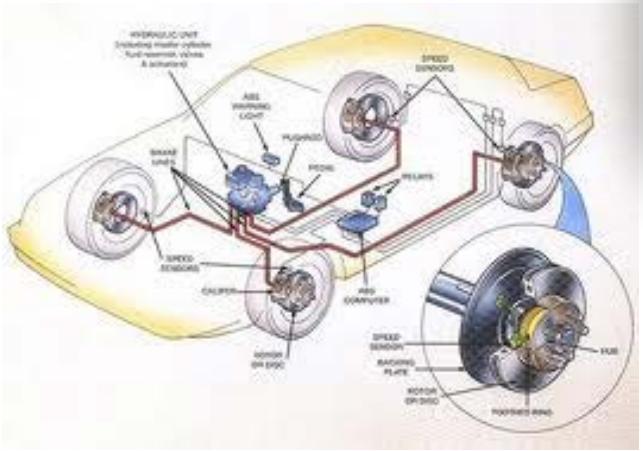
Research in software engineering.

- 9 faculty members
- ~ 40 researchers and engineers on projects

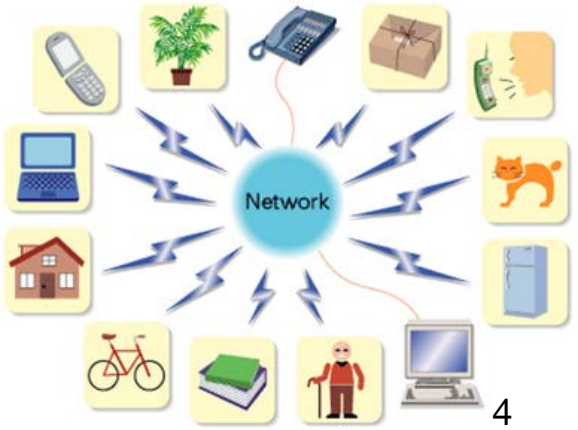
Triskell's Research Foundations / Topics



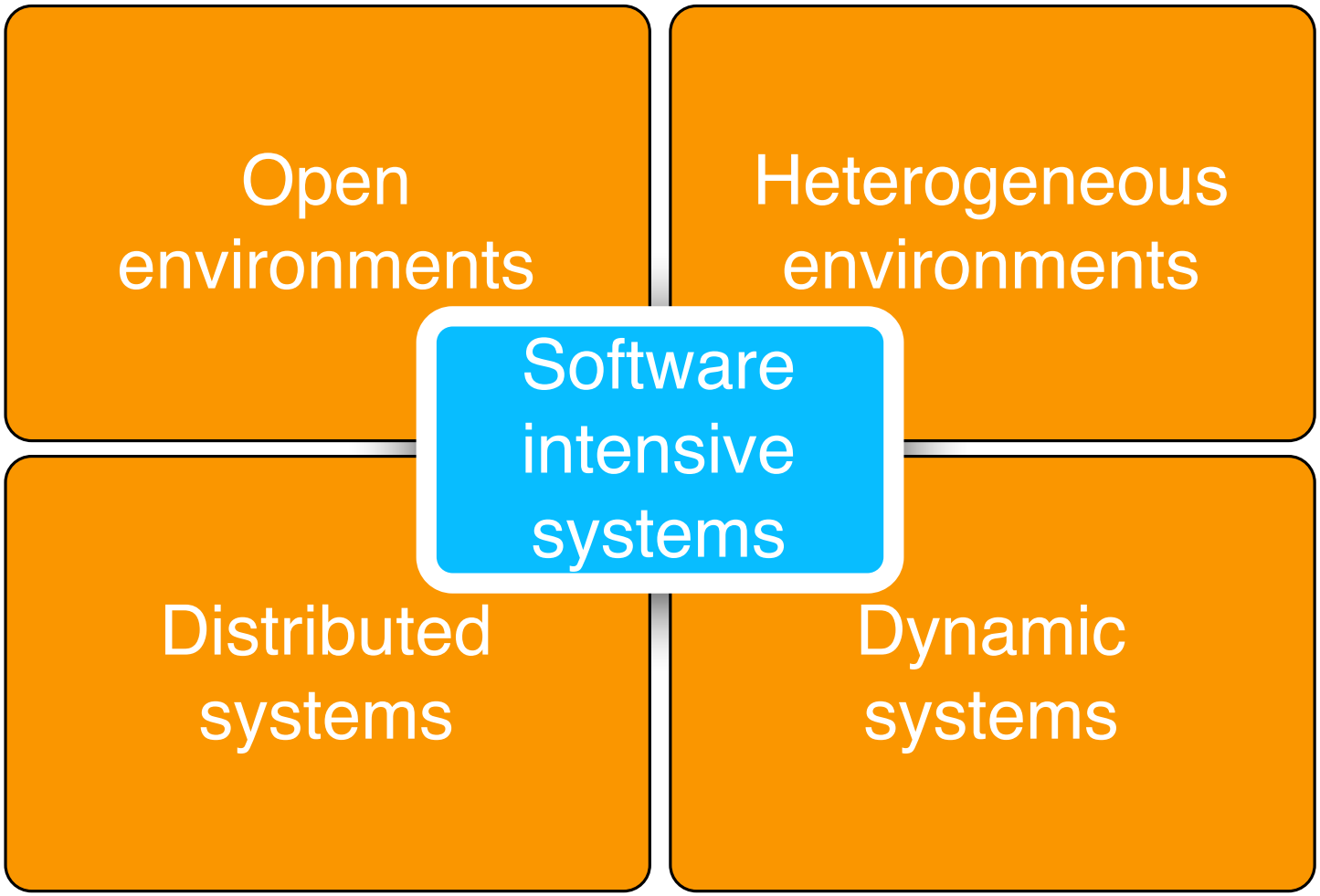
Application domains



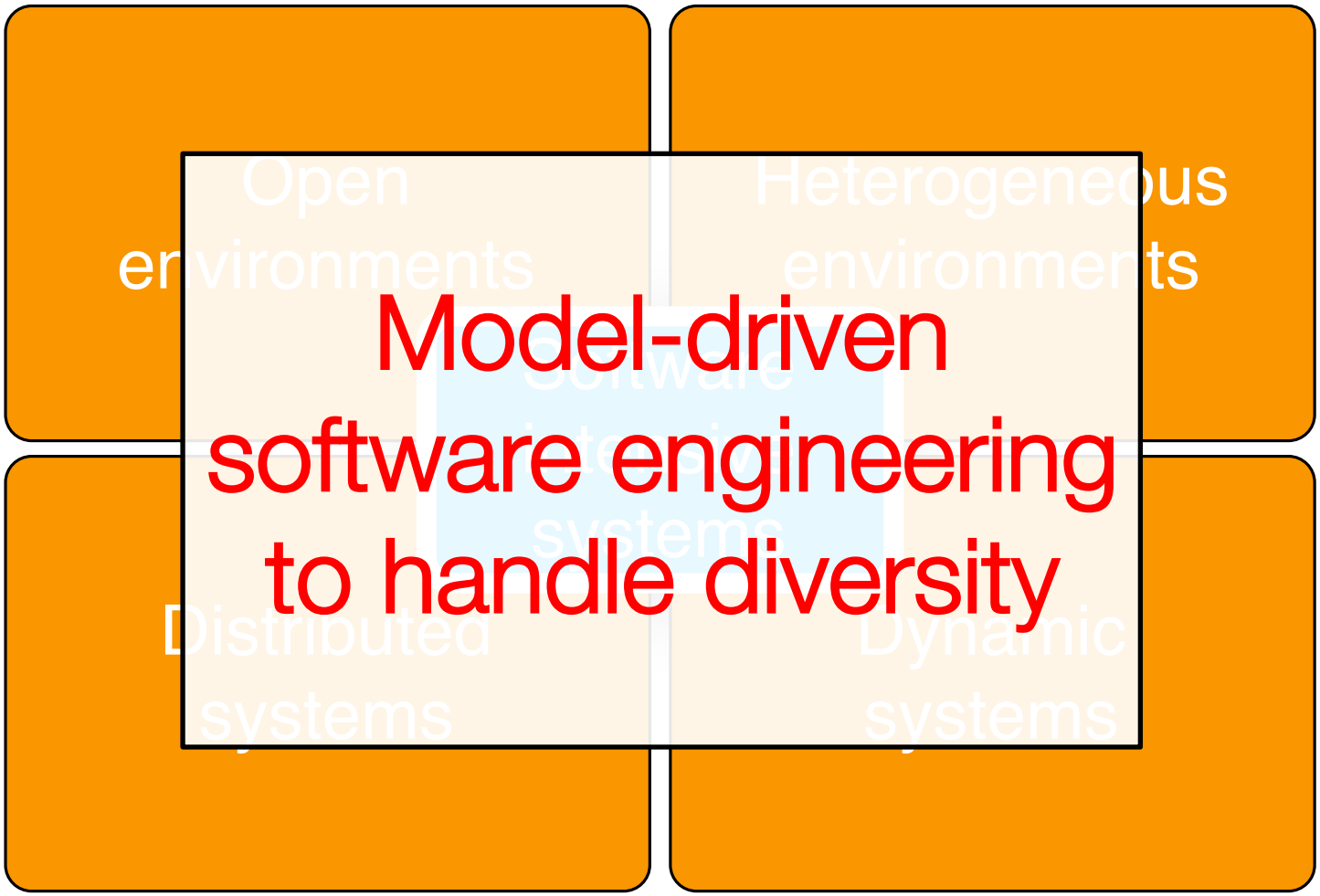
Software intensive systems



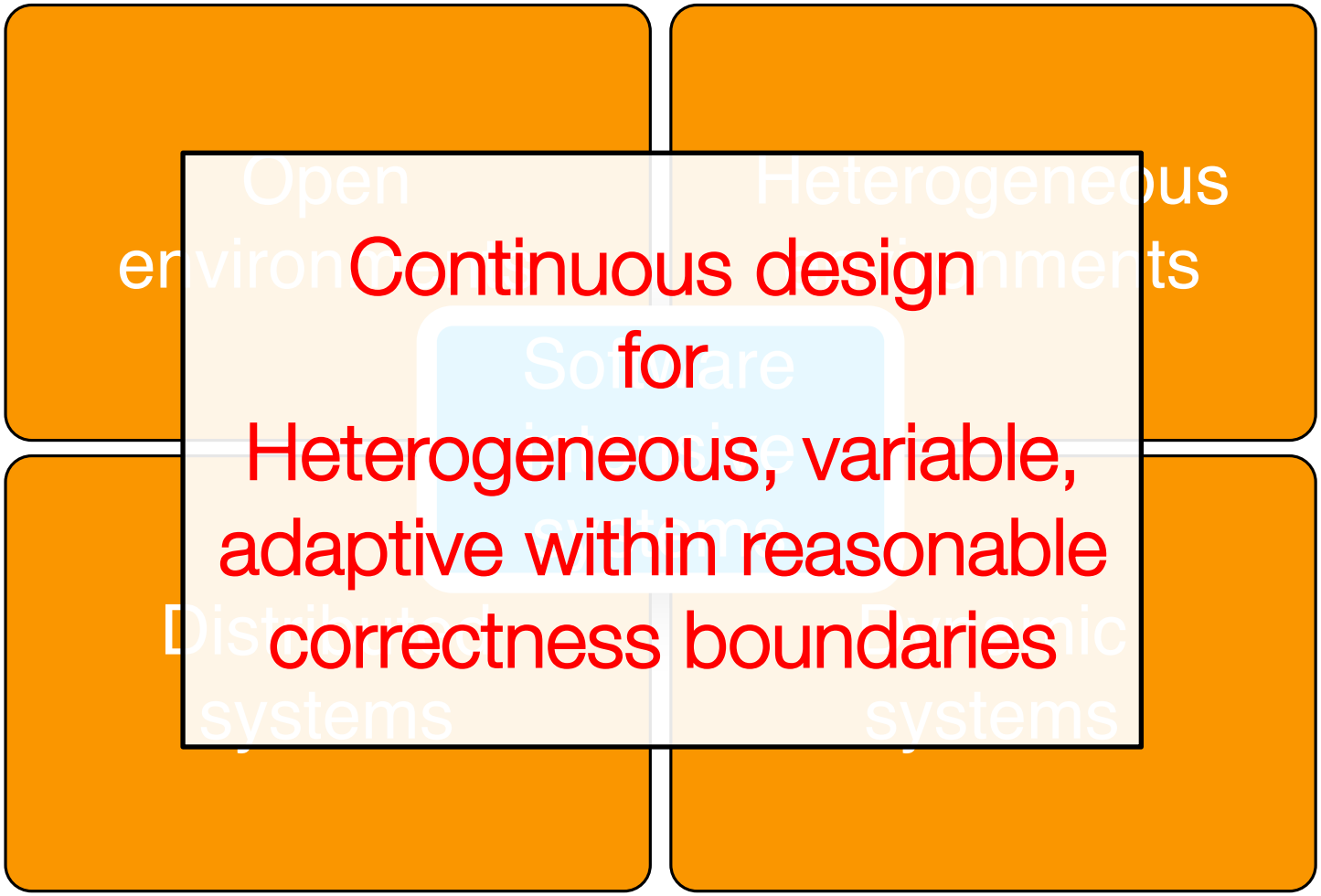
Context


Open environments
Heterogeneous environments
Software intensive systems
Distributed systems
Dynamic systems

Objectives


Open environments
Heterogeneous environments
Distributed systems
Dynamic systems
Model-driven software engineering to handle diversity

Approach


Open environments
Heterogeneous environments
Software intensive systems
Distributed systems
Dynamic systems
Continuous design
for
Heterogeneous, variable,
adaptive within reasonable
correctness boundaries

Aspect Oriented Model Driven Engineering

Distribution

Security

AOMDE = Pleonasm!

A model is an abstraction of an aspect of reality for a given purpose

"DSLs are far more prevalent than anticipated"

[Hutchinson *et al.*, ICSE'11]

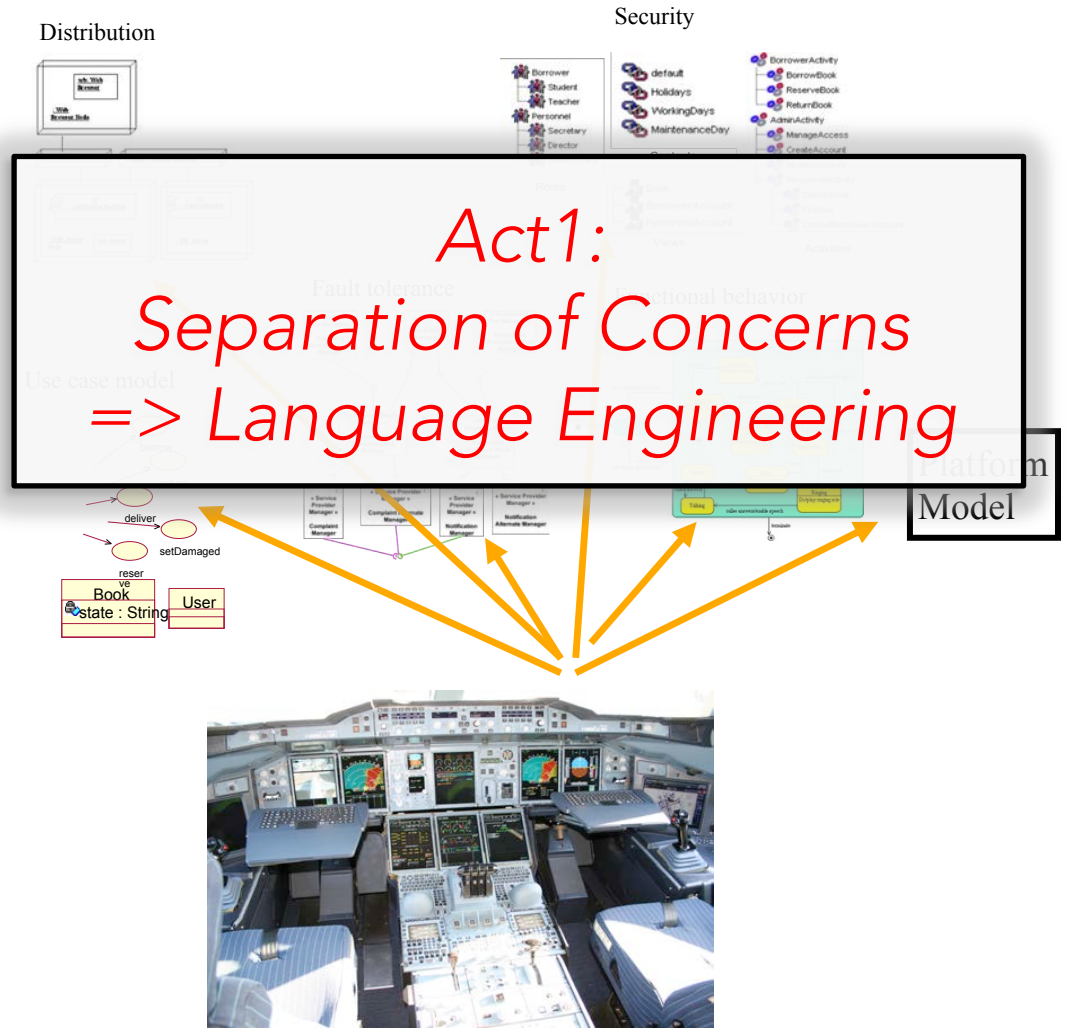
Platform Model

Design Model

Change one aspect and Automatically re-weave:
From Software Product Lines…
..to Dynamically Adaptive Systems

Code Model

Aspect Oriented Model Driven Engineering



Abstraction Gap

Problem Space

Assembler

C, Java

DSLs

orange™

ANDROID

Google

twitter

Solution Space

Domain-Specific (Modeling) Language ?

- "Language **specially** designed to perform a task in a **certain domain**"

- "A formal processable language targeting at a **specific viewpoint or aspect** of a software system. Its **semantics and notation** is designed in order to support working with that viewpoint as good as possible"

- "A computer language that's targeted to a particular kind of problem, **rather than a general purpose language** that's aimed at any kind of software problem."

GPL (General Purpose Language)

A GPL provides notations that are used to describe a computation in a human-readable form that can be translated into a machine-readable representation.

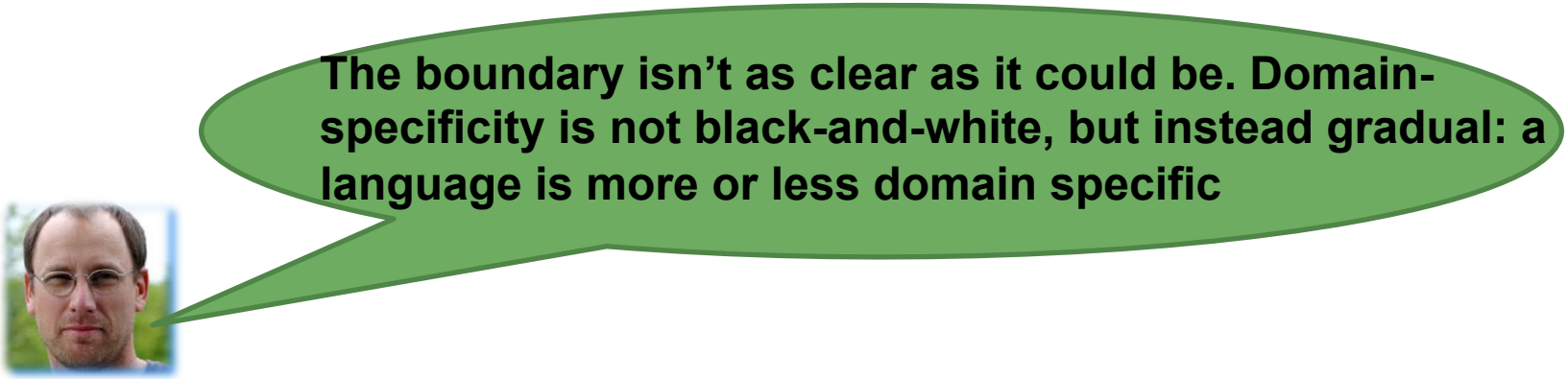
A GPL is a formal notation that can be used to describe problem solutions in a precise manner.

A GPL is a notation that can be used to write programs.

A GPL is a notation for expressing computation.

A GPL is a standardized communication technique for expressing instructions to a computer. It is a set of syntactic and semantic rules used to define computer programs.

GeneralPL vs DomainSL



	GPLs	DSLs
Domain	large and complex	smaller and well-defined
Language size	large	small
Turing completeness	always	often not
User-defined abstractions	sophisticated	limited
Execution	via intermediate GPL	native
Lifespan	years to decades	months to years (driven by context)
Designed by	guru or committee	a few engineers and domain experts
User community	large, anonymous and widespread	small, accessible and local
Evolution	slow, often standardized	fast-paced
Deprecation/incompatible changes	almost impossible	feasible

"Another lesson we should have learned from the recent past is that the development of 'richer' or 'more powerful' programming languages was a mistake in the sense that these baroque monstrosities, these conglomerations of idiosyncrasies, are really unmanageable, both mechanically and mentally.

aka **General-Purpose Languages**

I see a great future for very systematic and very modest programming languages"

aka **Domain-Specific Languages**

1972

ACM Turing Lecture, "The Humble Programmer"
Edsger W. Dijkstra

Empirical Assessment of MDE in Industry

John Hutchinson, Jon Whittle, Mark Rouncefield
School of Computing and Communications
Lancaster University, UK
+44 1524 510492
{j.hutchinson, j.n.whittle, m.rouncefield}@lancaster.ac.uk

Steinar Kristoffersen
Østfold University College and Møreforskning Molde AS
NO-1757 Halden
Norway
+47 6921 5000
steinar.kristoffersen@hiof.no

Model-Driven Engineering Practices in Industry

John Hutchinson
School of Computing and Communications
Lancaster University, UK
+44 1524 510492
{j.hutchinson@lancaster.ac.uk}

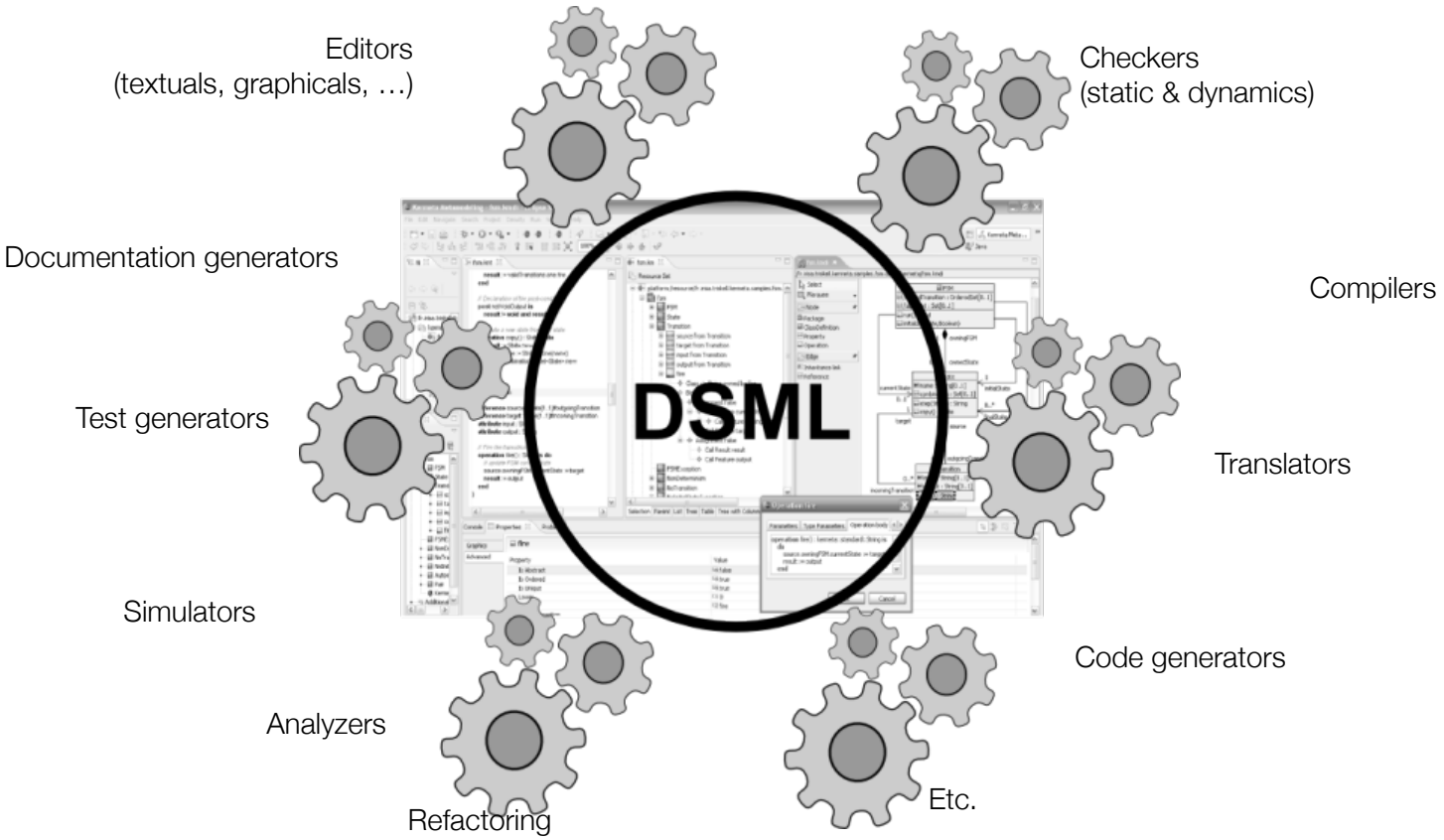
Mark Rouncefield
School of Computing and Communications
Lancaster University, UK
+44 1524 510492
{m.rouncefield@lancaster.ac.uk}

Jon Whittle
School of Computing and Communications
Lancaster University, UK
+44 1524 510492
{j.n.whittle@lancaster.ac.uk}

2011

"Domain-specific languages are far more prevalent than anticipated"

Domain-Specific (Modeling) Language



External DSLs *vs.* Internal DSLs

- An **external** DSL is a completely separate language and with its own custom syntax/tooling support (e.g., editor, compiler)

  Language worbenches: Eclipse modeling (EMF, xText, GMF, Sirius, Kermeta, xCore/xTend…), DSL Tools, Meta Edit+, MPS, GME, Neverlang, Delite, etc.

- An **internal** DSL is more or less a set of APIs written on top of a host language

  Extension mechanisms: xTend's active annotation, Scala's LMS, extensions methods (e.g., xTend, Kotlin, Scala-Virtualized), plain-old java annotation or even fluent interfaces!

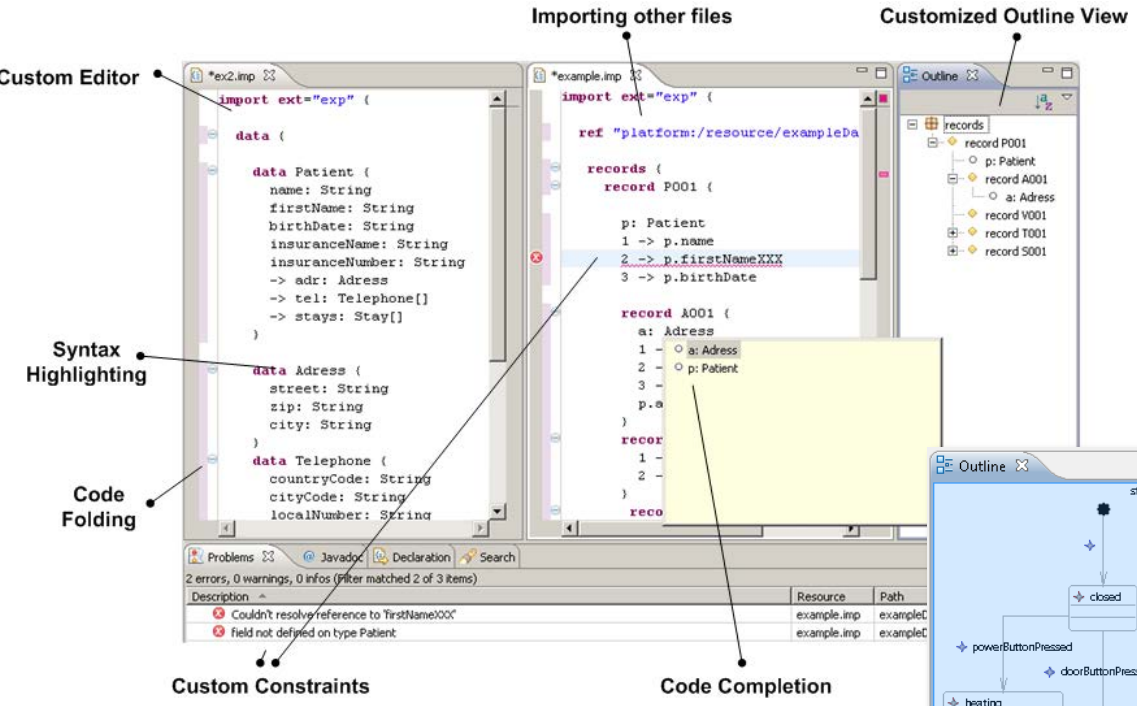
⇒ DSL technology is here (no excuse)

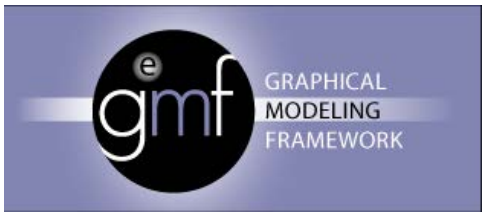
⇒ *Related fields: generative programming, product lines*

External DSLs *vs.* Internal DSLs

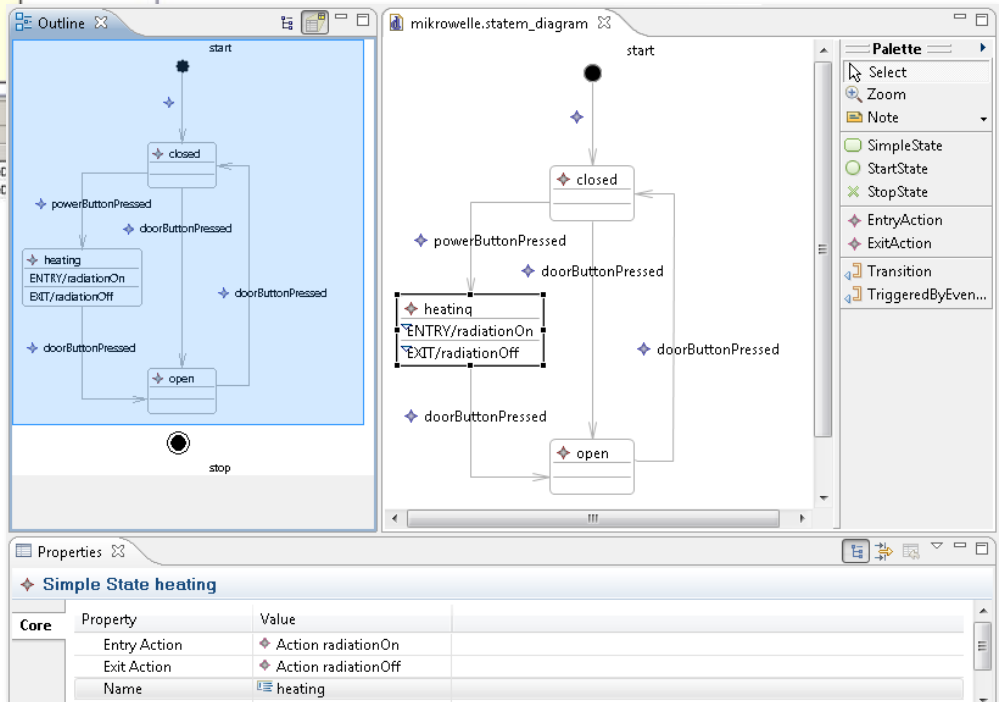
- Both internal and external DSLs have strengths and weaknesses
  - learning curve,
  - cost of building,
  - programmer familiarity,
  - communication with domain experts,
  - mixing in the host language,
  - strong expressiveness boundary

Textual DSLs *vs.* Graphical DSLs

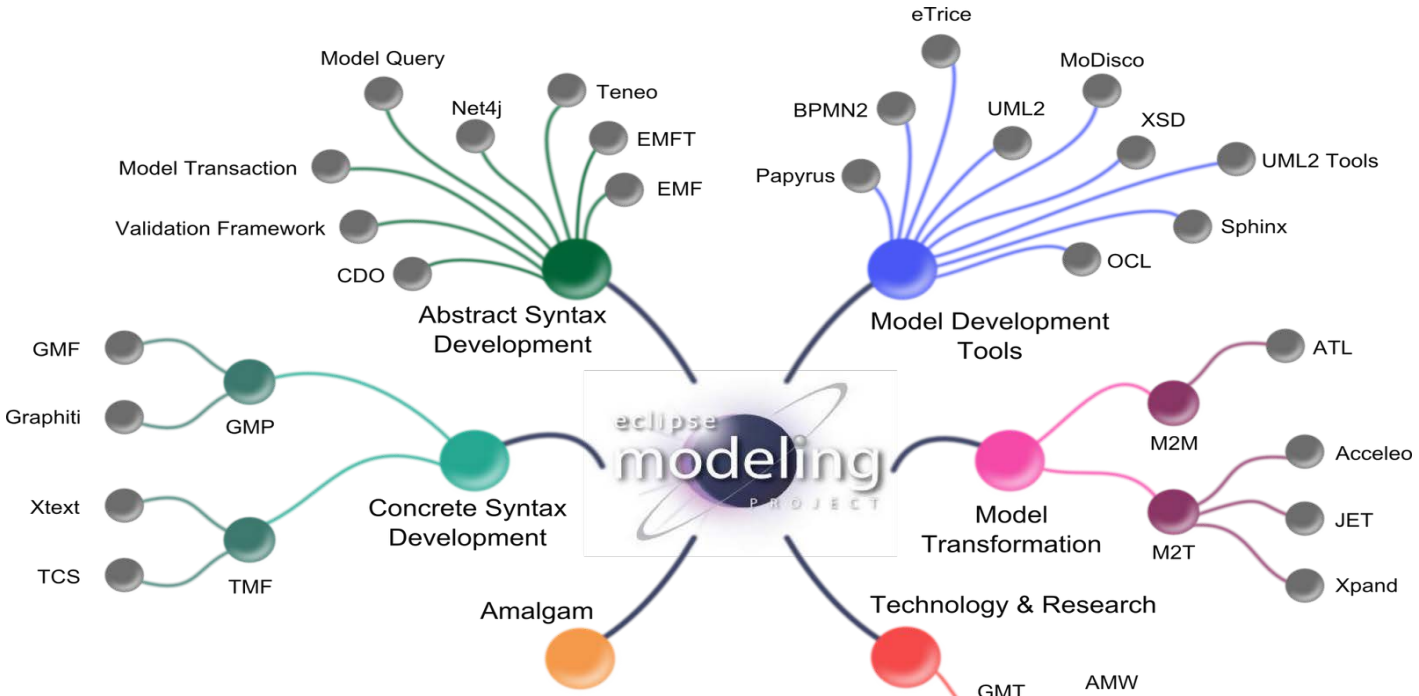


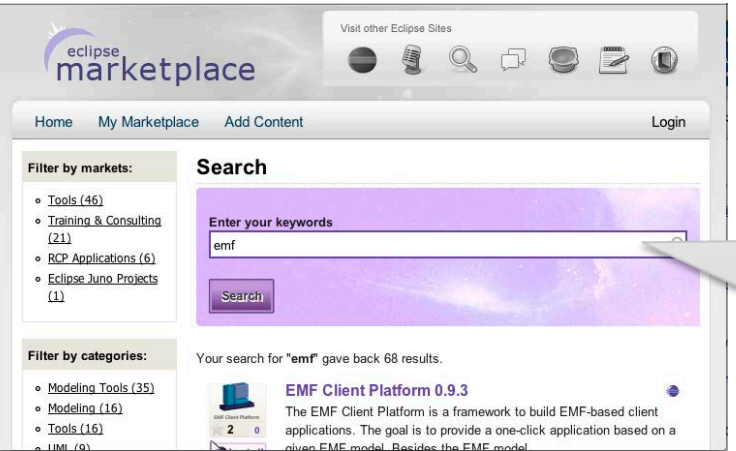


xtext



Eclipse Modeling





EMF-based Tools *(68 matches in September, 2012)*

The Kermeta Language Workbench [SoSyM'13]

- **Modular design of DSMLs**
  - One meta-language per language concern (`require`)
    - Ecore, OCL, Kermeta Action Language
    - But also: QVTo, fUML, Alf, Ket, Xsd…
  - Static introduction mechanism (`aspect`)
- **Efficient implementation of DSMLs**
  - Mashup of the meta-languages to efficient bytecode
  - Integrated with third-party tools (EMF compliant)
- *Current investigations: precise metamodeling, modeling in the large, language family, multi-platforms, model comprehension…*

Aspect Oriented Model Driven Engineering

Distribution

Security

Act2:
Composition of Concerns
=> Global Engineering

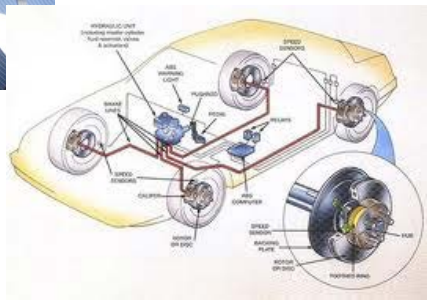
Platform Model

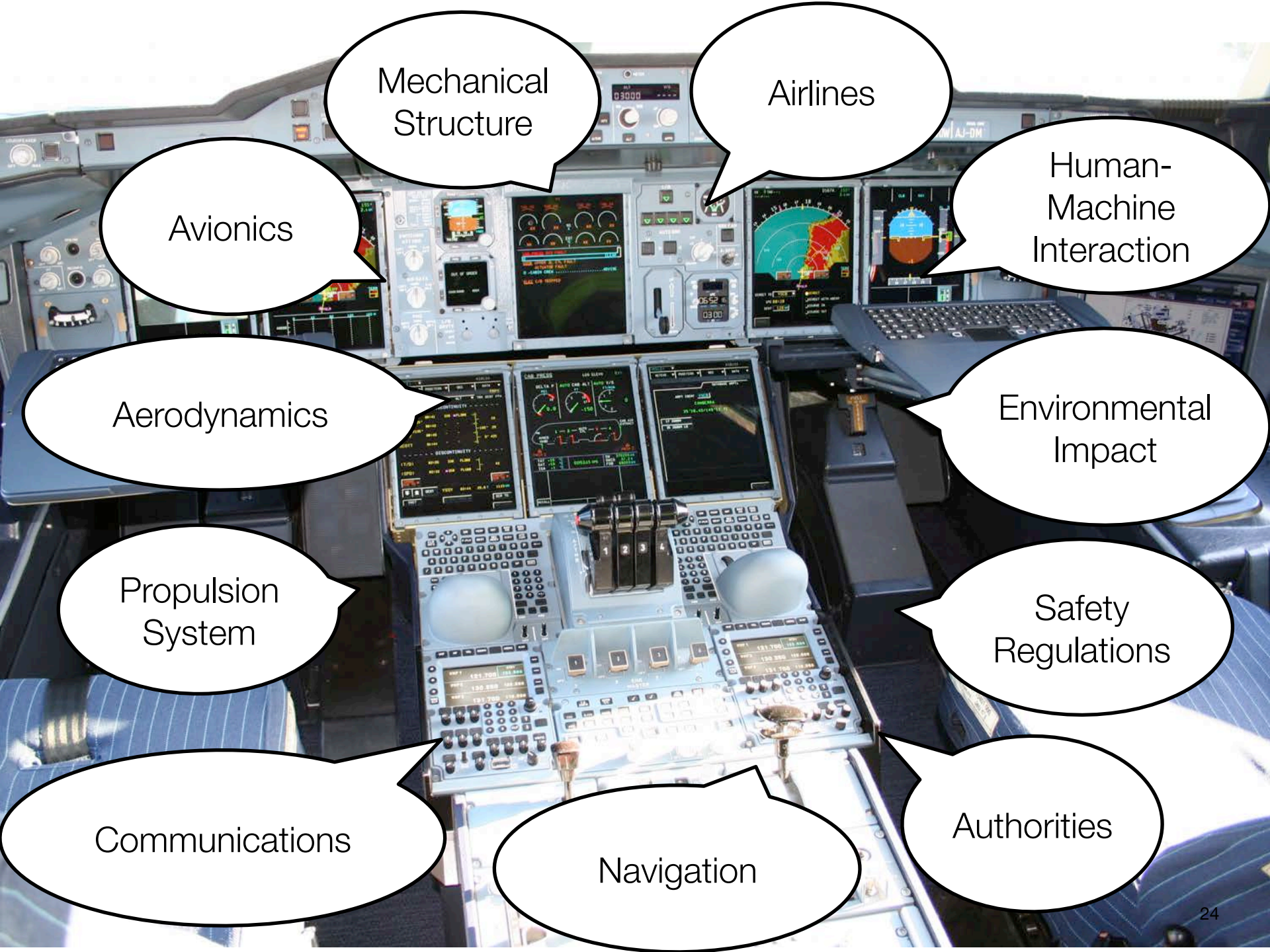
Design Model

Code Model

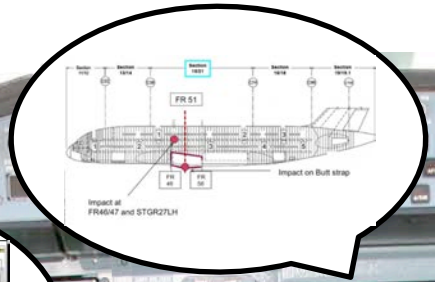
Complex Software-Intensive Systems

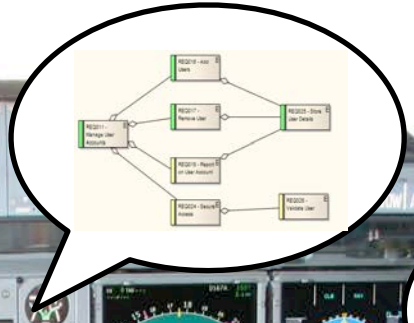
- deal with multiple concerns
  ⇒ require *global analysis and execution*
- integrate heterogeneous parts
  ⇒ require *global service*
- manage evolution of concerns and the emergence of new concerns
  ⇒ require evolution and creation of tools and methods for software development

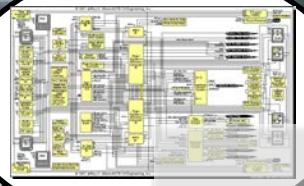


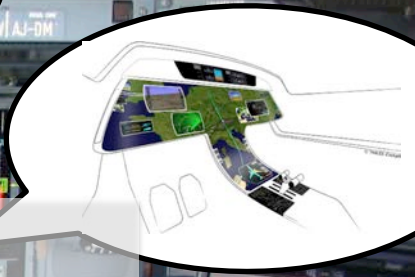

Mechanical Structure
Airlines
Avionics
Human-Machine Interaction
Aerodynamics
Environmental Impact
Propulsion System
Safety Regulations
Communications
Navigation
Authorities

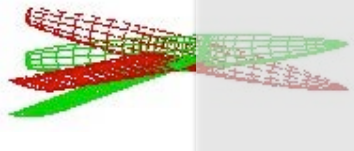
Heterogeneous Domain-Specific Modeling Languages

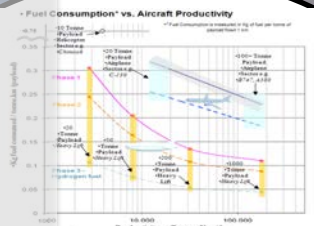


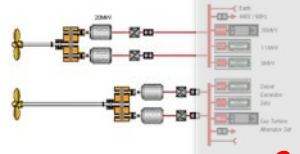


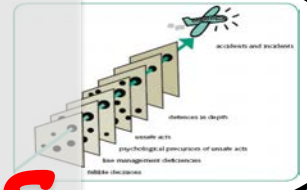


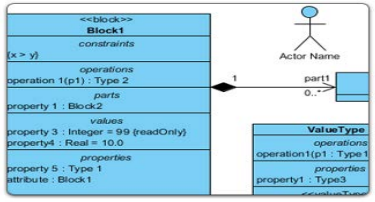


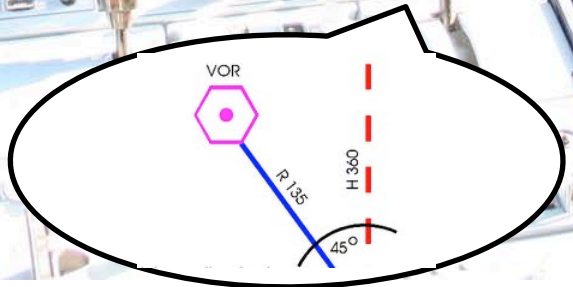












(AO) MDE

=

(AO) Modeling

+

Composition

Why? *various intents*
When? *design vs. run time*
What? *homogeneous vs. Heterogeneous*
Where? *static vs dynamic introduction*
How? *symmetric vs. Asymmetric*

=> different techniques for composition

Challenges

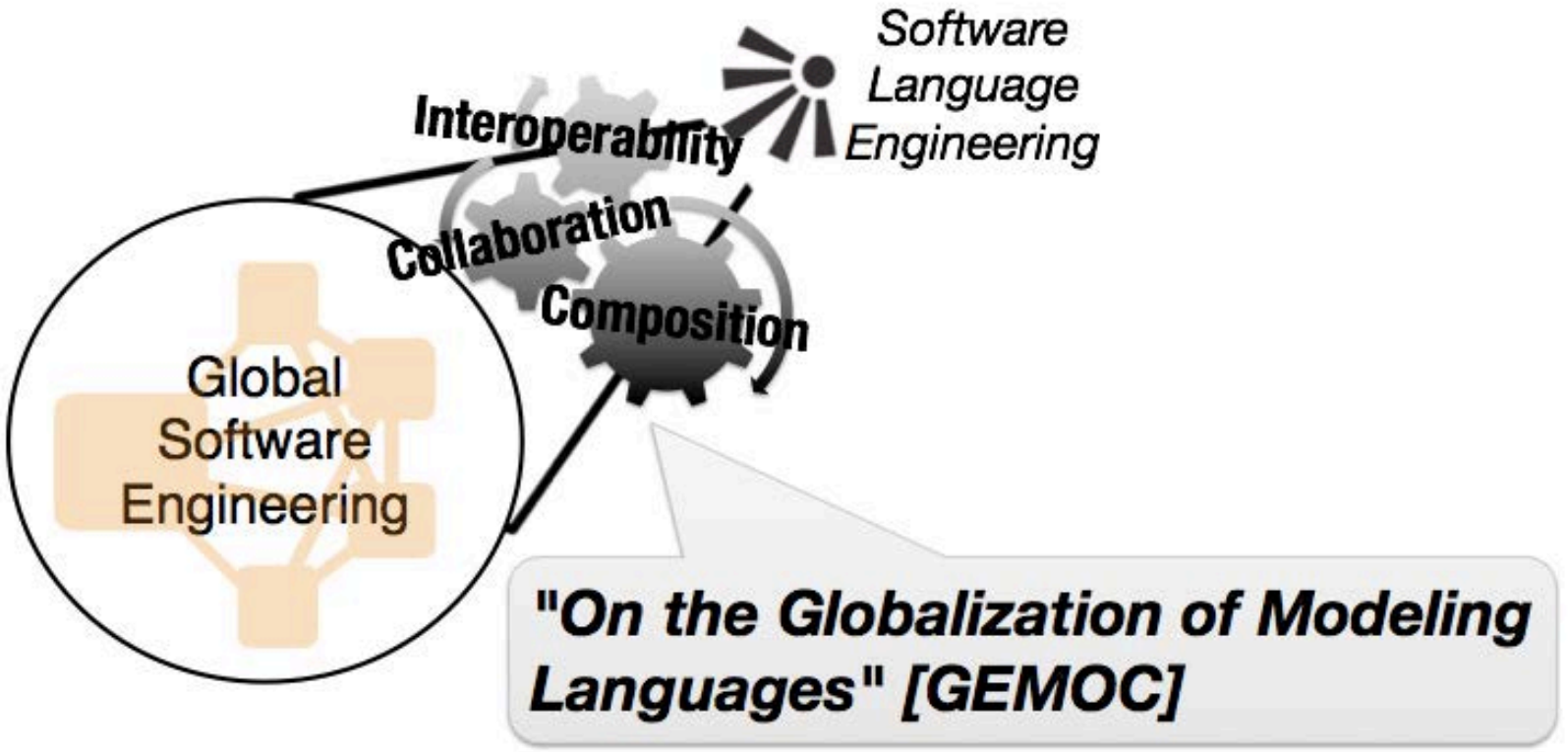
- *Model (Driven)* Engineering
  - ➔ *(Software) Language* Engineering
  - ➔ *Global (Software)* Engineering

- Language relationships should be capitalized
  - ⇒ from transformation to composition
- Global model coordination and analysis
  - ⇒ from design to runtime


Gemoc
On the Globalization
of Modeling Languages!

An Initiative…

Focuses on SLE tools and methods for interoperable, collaborative, and composable modeling languages



… Constantly Growing

The GEMOC Initiative *is born!*

An open initiative to

- coordinate (between members)
- disseminate (on behalf the members)

worldwide R&D efforts on the globalization of modeling languages

http://gemoc.org

- *"Supporting coordinated use of DSMLs leads to what we call **the globalization of modeling languages**, that is, **the use of multiple modeling languages to support coordinated development** of diverse aspects of a system."*

- Advisory Board: Benoit Combemale (Fr.), Robert B. France (USA), Jeff Gray (USA), Jean-Marc Jézéquel (Fr.)
- Funded by complementary and successive projects (IP left to PCA of each projects)

The GEMOC Initiative: Objectives

Globalized DSMLs aim to support the following critical aspects of developing complex systems:

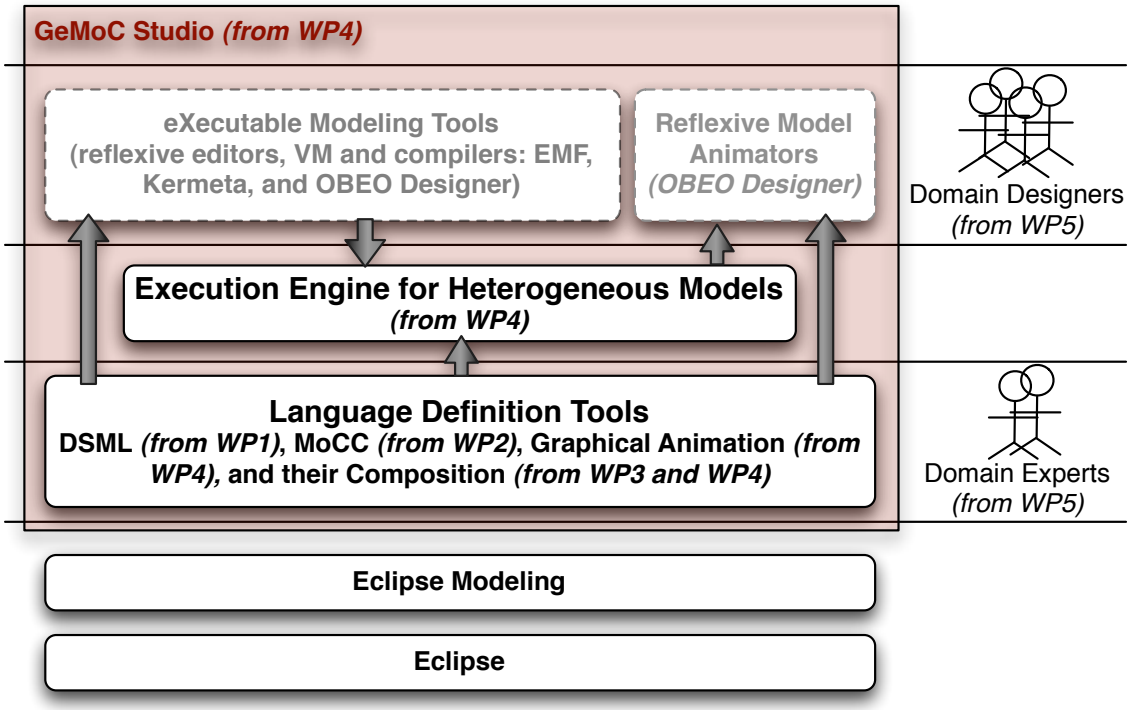
- **communication** across teams working on different aspects,
- **coordination** of work across the teams,
- and **control** of the teams to ensure product quality.

Current investigated application domains:

- Complex software-intensive systems: safety-critical embedded systems, cyber-physical systems, system of systems, dynamic adaptable systems
- Enterprise Architecture

The GEMOC Studio

- A *language* workbench for domain experts
  - DSMLs implementation and coordination
- A *modeling* workbench for domain designers
  - Heterogeneous modeling and simulation



ANR INS GEMOC (2012-2016)

"A Language Workbench for Heterogeneous Modeling and Analysis of Complex Software-Intensive Systems »

Tools and methods for the definition and coordination of *heterogeneous* executable *modeling languages* over *heterogeneous models of computation*

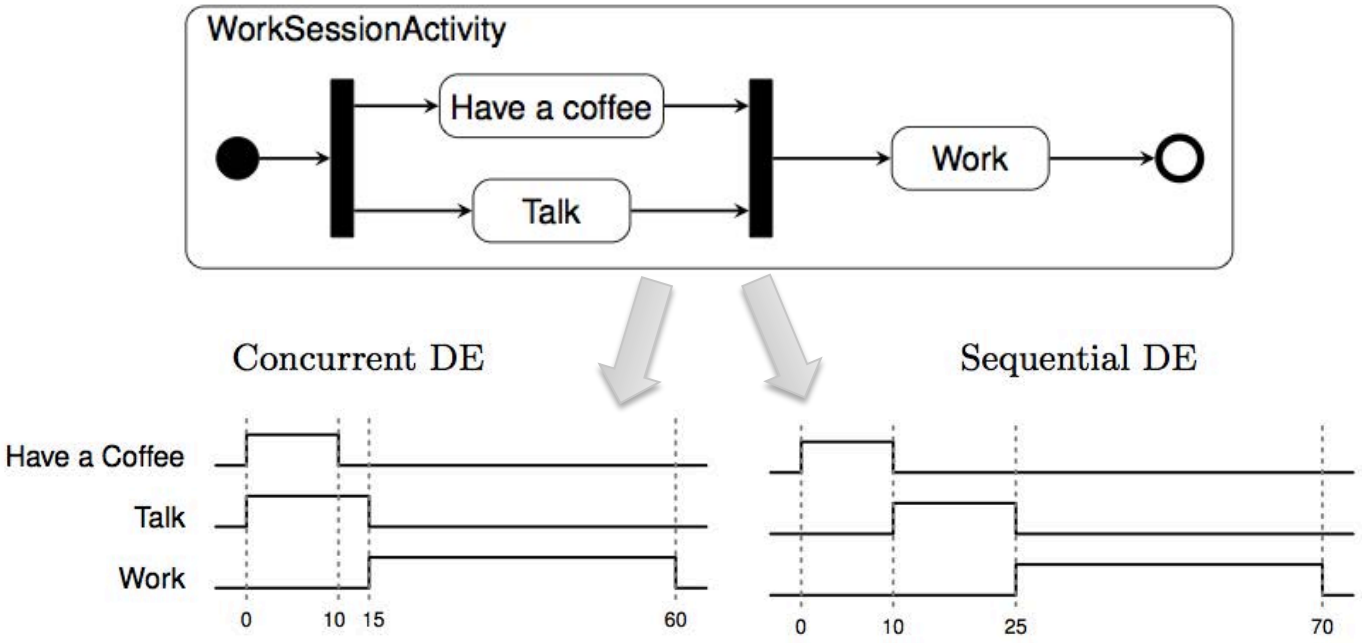
http://gemoc.org/ins

Concurrent execution of heterogeneous domain-specific models

- Metamodeling: Effective environments for the design and implementation of executable domain specific languages (*e.g.,* Kermeta at Inria)
  - BUT these environments do not allow the integration of heterogeneous models of computation (concurrence, communication…)
- Models of computation: Effective environments to deal with the execution and analysis of models based on heterogeneous models of computation (*e.g.,* Ptolemy at UC Berkeley, ModHel'X at Supélec)
  - BUT these environments do not allow adaptation to specific business/application domains

Bridging the gap between *language theory* and *concurrency theory*

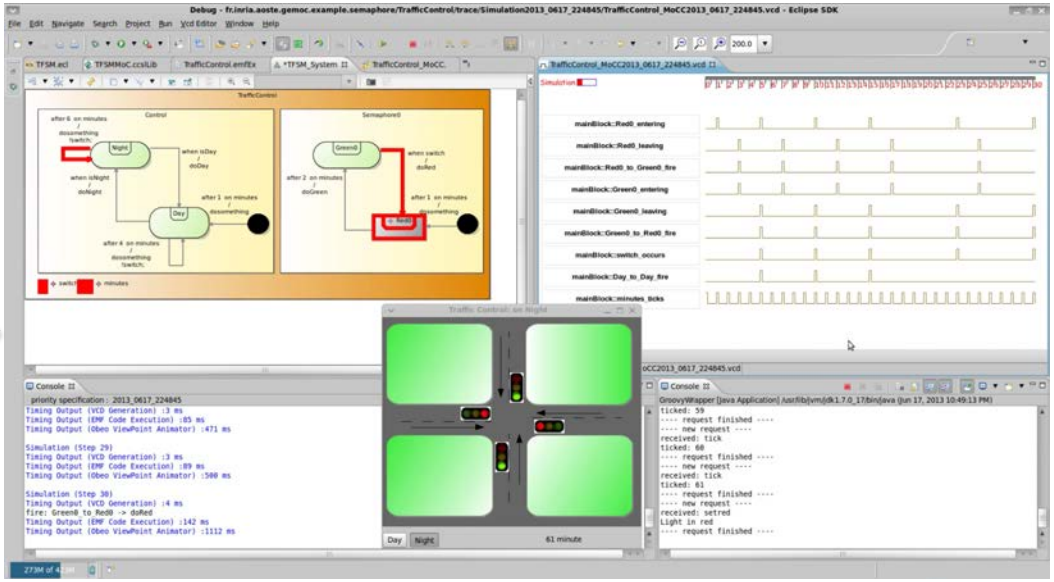
fUML Implementation



Bridging the Chasm between Executable Metamodeling and Models of Computation (Benoit Combemale, Cécile Hardebolle, Christophe Jacquet, Frédéric Boulanger, Benoit Baudry), In 5th Int'l Conference on Software Language Engineering (SLE), 2012.

Concurrent execution of *homogeneous* domain-specific models

Timed Finite State Machine (TFSM) Implementation



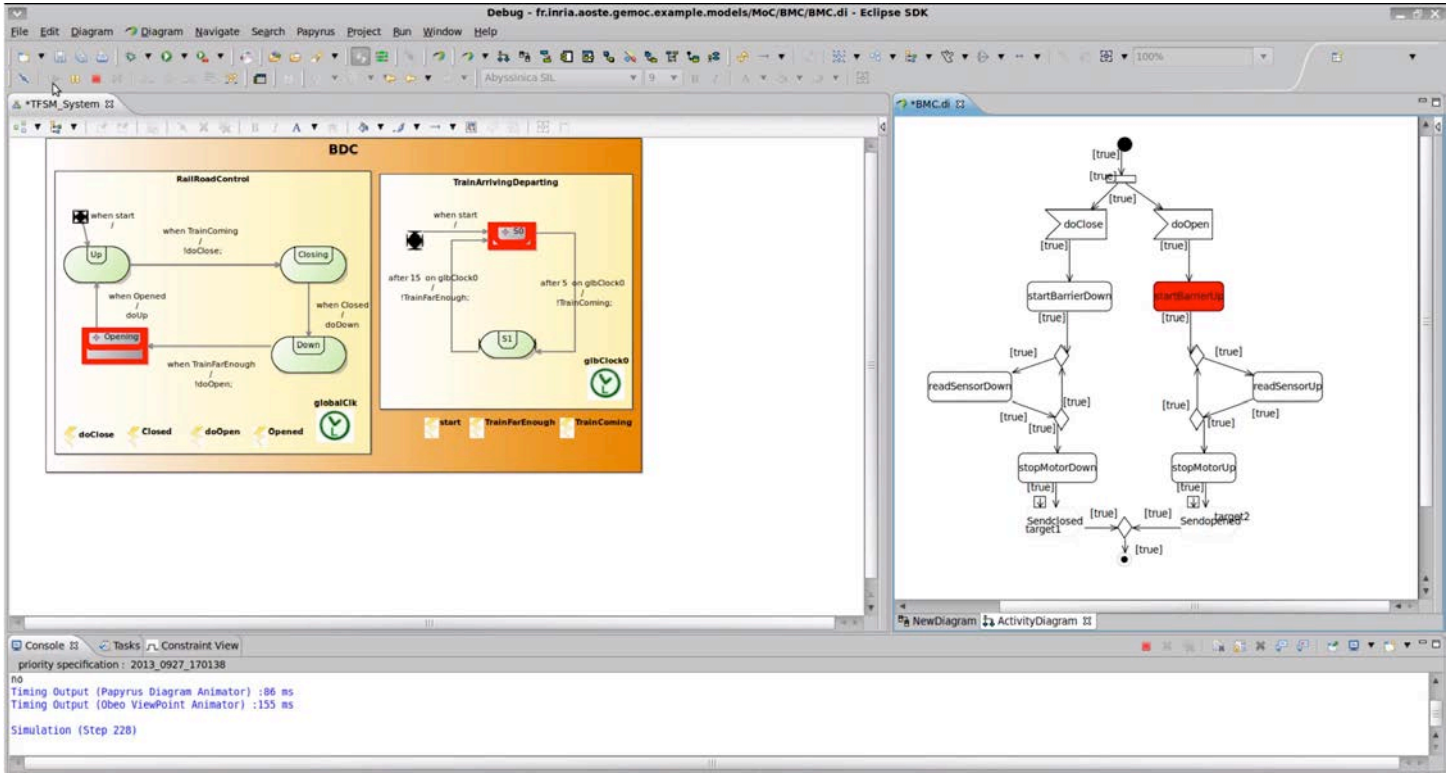
… but also Logo, Actor model

Companion webpage: http://gemoc.org/sle13

Reifying Concurrency for Executable Metamodeling (Benoit Combemale, Julien Deantoni, Matias V. Larsen, Frédéric Mallet, Olivier Barais, Benoit Baudry, Robert B. France), In 6th Int'l Conference on Software Language Engineering (SLE), 2013.

Concurrent execution of *heterogeneous* domain-specific models

fUML and TFSM Coordination



Railroad Crossing Heterogeneous Model (Matias Vara Larsen, Arda Goknil), In 1st Int'l Workshop on The Globalization of Modeling Languages (GEMOC workshop), 2013.

The GEMOC Studio: Current Status

- A workbench based on EMF
  - A language workbench for DSML Engineers
    - design and implement executable domain specific modeling languages with explicit model of computation, and (structural and behavioral) language interfaces
    - define structural and behavioral relations between DSMLs
  - A modeling workbench System/Software Engineers
    - model the heterogeneous aspects of complex software-intensive systems using various DSML
    - simulate and animate the coordinated execution of the heterogeneous models
- Current integrated technologies:
  - EMF, incl. Ecore, Ecore tools, and OCL
  - Kermeta, xTend, CCSL, Cometa and ECL
  - xText, Obeo Designer (Sirius)
  - GEMOC Execution Engine, Timesquare, Obeo Animator

Roadmap (ANR GEMOC)

- Language workbench
  - Explicit DSML interface
  - Meta-language for DSMLs coordination
  - Methodology for designing DSMLs and their coordination
- Heterogeneous model execution
  - Execution trace
  - Graphical animator design
  - Combining *continuous* time and *discrete* time

Conclusion and Perspectives

- Agile (Software Development)
  vs. (Agile Software) Development

- *Model Driven* Engineering
  ⇒ *Software Language* Engineering
  ⇒ *Globalization* of Modeling Languages

- The GEMOC Initiative

GEMOC Initiative: http://gemoc.org

GEMOC Studio: https://ci.inria.fr/gemoc

Contact:

Benoit Combemale

benoit.combemale@irisa.fr

http://combemale.fr