CAMPaM 2012 Introduction

Frédéric Boulanger   Cécile Hardebolle   Christophe Jacquet

Supélec E3S – Computer Science Department

April 2012

2012-04-28 18:33

Agenda

1. Who are we?
2. Our research interests
3. Topics of interest for this week
4. Criteria of success

Who are we?

Supélec

- French engineering school (“grande école”)
- 460 engineering degrees delivered each year
- Continuing education

Supélec Systems Science

Multi-disciplinary research team

- Power systems, Control Science, Electro-magnetism, Telecommunications, Signal processing
- ... and Computer science

Computer Science Department

- Personalization of hypermedia and web queries
- Optimization of high-performance networks
- Modeling and validation of heterogeneous systems

Our View of Heterogeneity

System Design

Our View of Heterogeneity

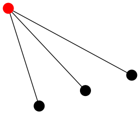
System Design

Starting from an overall specification

Our View of Heterogeneity

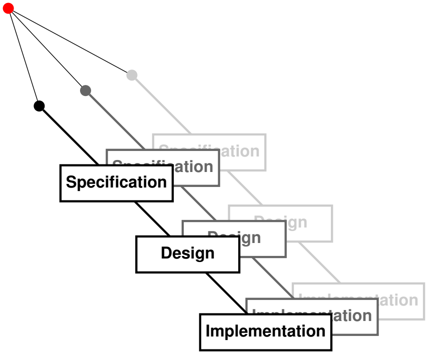
System Design

Decompose into simpler subsystems



System Design

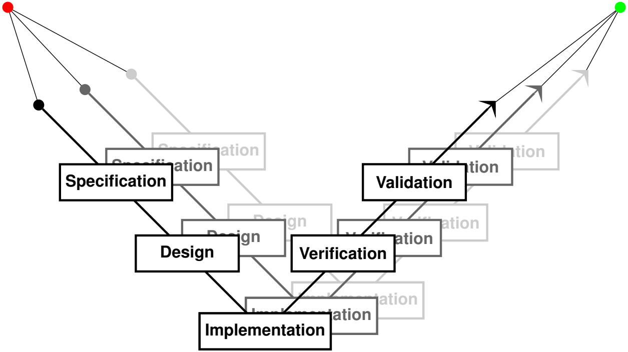
Refine down to implementation



Our View of Heterogeneity

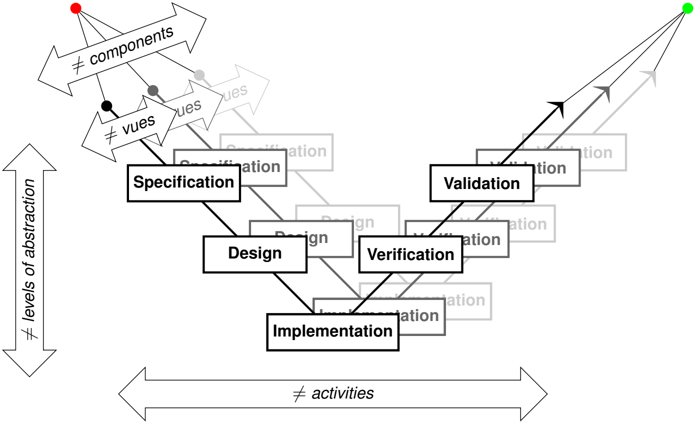
System Design

Recompose components to build the system (integration)



Our View of Heterogeneity

Four sources of heterogeneity

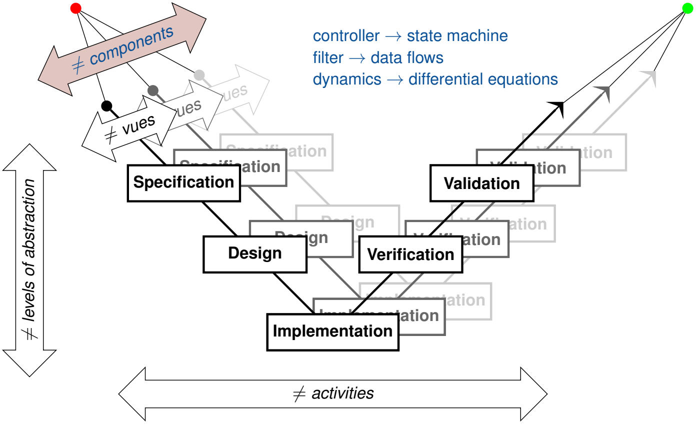


Our View of Heterogeneity

1 - Different components are modeled according to different paradigms

⇒ Issue: combine heterogeneous behaviors

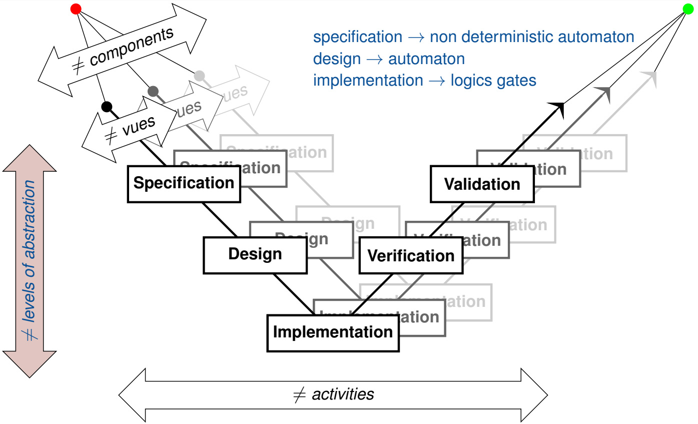
controller → state machine
filter → data flows
dynamics → differential equations



Our View of Heterogeneity

2 - Different formalisms are used at different levels of abstraction

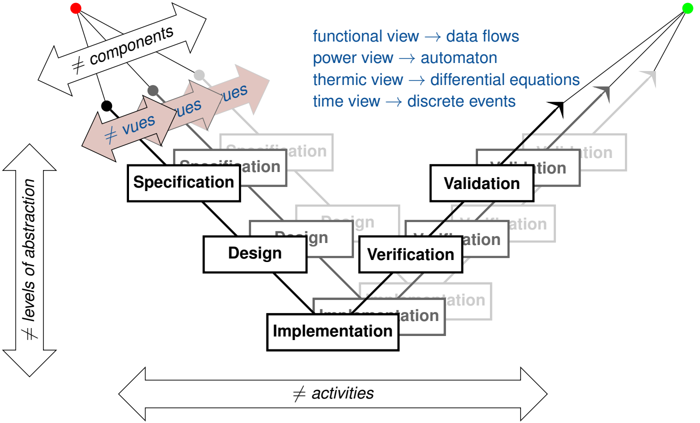
$\Rightarrow$ Issue: conformance of refinements



Our View of Heterogeneity

3 - Different views of a component use different paradigms

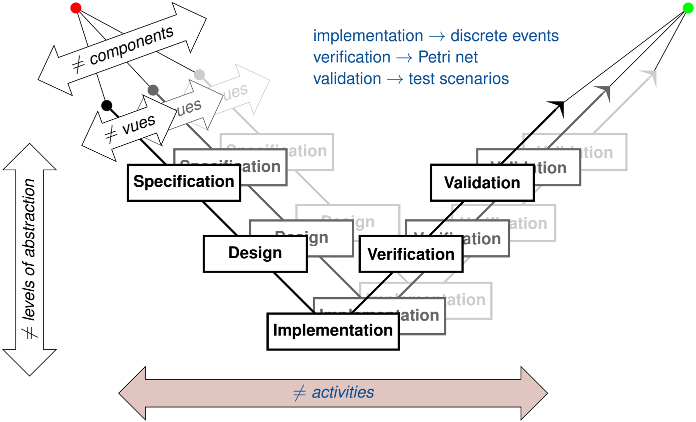
⇒ Issue: synchronize heterogeneous models



Our View of Heterogeneity

4 - Different activities require different formalisms

⇒ Issue: consistence of different models



Problems Raised by Heterogeneity

Four problems to solve

1. Composition of components ⇒ combine heterogeneous behaviors
   ModHel'X Interface blocks, Semantic adaptation
2. Different levels of abstraction ⇒ conformance of refinements
   Conformance testing, B method
3. Superposition of views ⇒ synchronize heterogeneous models
   VUML (OCL constraints)
4. Per activity models ⇒ consistence of different models
   Validation of model transformations

Transverse Issues

- Extra-functional views constrain the refinement of the functional views
- Decomposition is not necessarily the same in all views/activities
  (no one-to-one mapping between components of different views)

Problems Raised by Heterogeneity

Four problems to solve Our current work

1. Composition of components ⇒ combine heterogeneous behaviors
   ModHel'X Interface blocks, Semantic adaptation
2. Different levels of abstraction ⇒ conformance of refinements
   Conformance testing, B method
3. Superposition of views ⇒ synchronize heterogeneous models
   VUML (OCL constraints)
4. Per activity models ⇒ consistence of different models
   Validation of model transformations

Transverse Issues

- Extra-functional views constrain the refinement of the functional views
- Decomposition is not necessarily the same in all views/activities
  (no one-to-one mapping between components of different views)

Execution of heterogeneous models

Models of computation

- Set of rules for composing the behavior of components
- Defines the semantics of the structure of a model
- Modeling languages described as MoCs $\Rightarrow$ common syntax

ModHel'X

- Generic meta-model for describing models
- Generic execution engine for executing models according to any MoC

Semantic adaptation

- Hierarchical heterogeneity
- Semantic adaptation between two heterogeneous models
  - Adaptation of data
  - Adaptation of control
  - Adaptation of time

Modular Modeling Language Definition

Build custom modeling languages by assembling off-the-shelf aspects:

- Structure (meta-model, data types)
- Elementary behaviors (Turing machines, state machines, Petri nets)
- Control (instantaneous broadcast, (a)synchronous communications)
- Time (discrete, continuous, with duration, triggering)

Multi-Purpose Modeling Language Definition

Most modeling languages definitions are specific to an activity
A multi-purpose modeling language definition should be usable for:

- Simulation
- Code generation
- Glue code generation for adaptation
- Verification

Criteria for success

- Have a nice week ;-)
- Create connections between teams working on similar topics
- Start new projects
- Publish joint paper(s)