Taming Dynamically Adaptive Systems using Models and Aspects

Brice Morin, Olivier Barais, Grégory Nain
and Jean-Marc Jézéquel

UNIVERSITÉ DE RENNES 1

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

INRIA

DiVA: European Project on Dynamic Variability in Complex Adaptive Systems
http://www.ict-diva.eu/

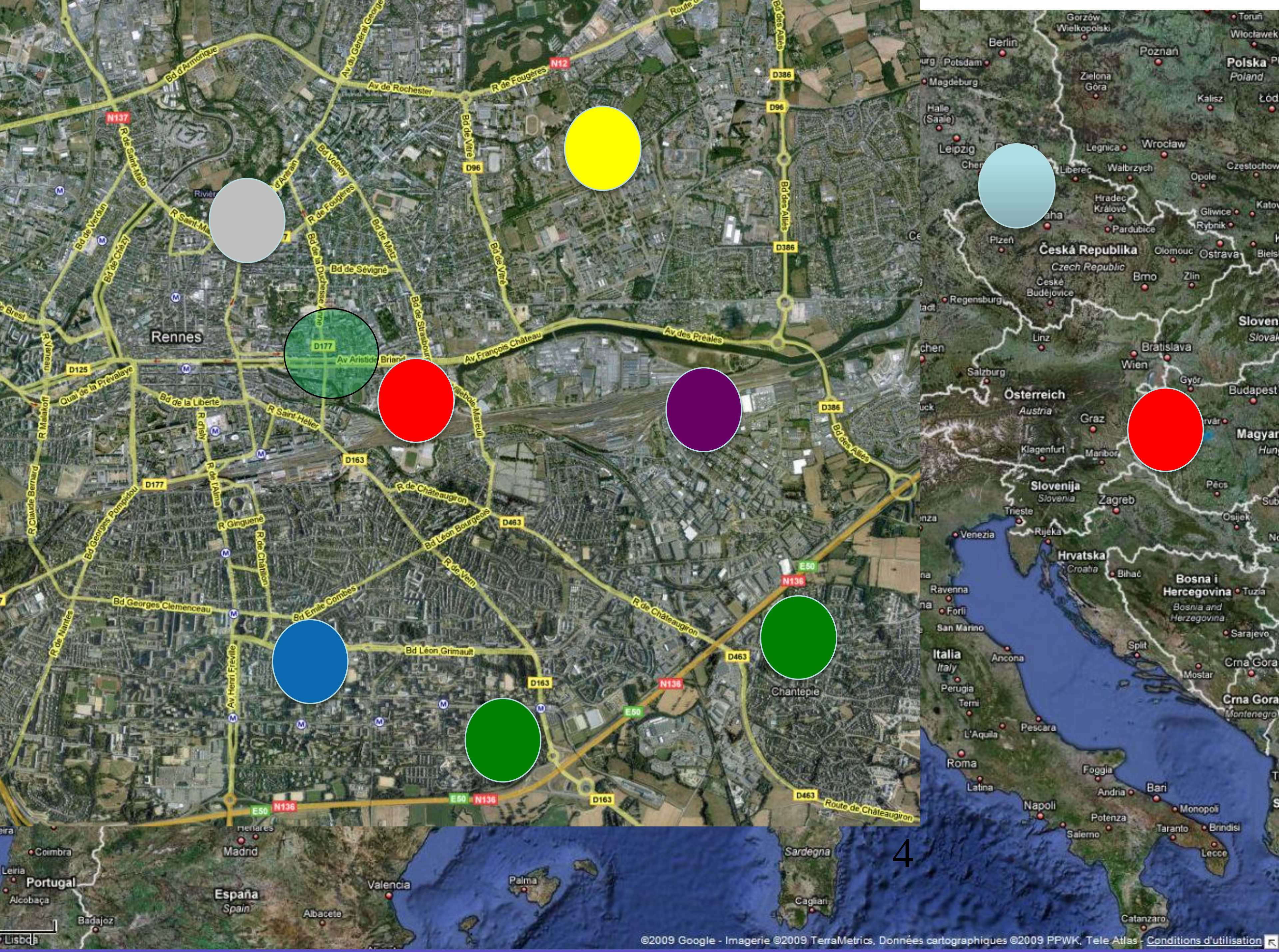
S-Cube: European Network of Excellence on Software Services and Systems
http://www.s-cube-network.eu/

Outline

- Context, Problems and Related Work
- Solutions to meet the challenges
- Conclusion
- Future works

Context

- Home-automation to help disabled people to stay at home
  - Aging society
  - Hospital have limited resources, rooms, etc
    - → Very short stays
  - Long stays very expensive for people and society
  - Houses, flats, etc should be equipped


Rennes
Av Aristide Briand
Av François Château
Av des Préales
Bd de Sévigné
Bd Léon Grimault
Bd Georges Clemenceau
Bd Emile Combes
Bd Léon Bourgeois
R de Châteaugiron
Route de Châteaugiron
Bd des Alliés
Bd de Vitré
Av de Rochester
R de Fougères
Bd d'Armorique
Bd de Verdun
Bd de la Liberté
Quai de la Prévalaye
R Saint-Hélier
R de Vern
Av Henri Fréville
Bd Georges Pompidou
Chantepie
N12
N136
N137
E50
D96
D125
D163
D177
D386
D463
Berlin
Poznań
Polska
Poland
Wrocław
Leipzig
Česká Republika
Czech Republic
Brno
Ostrava
Bratislava
Wien
Österreich
Austria
Graz
Budapest
Magyar
Slovenija
Slovenia
Zagreb
Hrvatska
Croatia
Bosna i Hercegovina
Bosnia and Herzegovina
Sarajevo
Italia
Italy
Venezia
Roma
Napoli
Bari
Madrid
España
Spain
Valencia
Portugal
Sardegna
©2009 Google - Imagerie ©2009 TerraMetrics, Données cartographiques ©2009 PPWK, Tele Atlas - Conditions d'utilisation

Many Different Needs 1/2

Mrs. Dupont

Picture from http://www.apt.gc.ca

- Living at home
- Motion troubles
- Memory loss
- Speaks French (only)
- Home equiped with :
  - LonWorks (lights)
  - Velux (shutters)

Many Different Needs 2/2

Picture from http://ditwww.epfl.ch

Mr. John Doe

- English student
- Living at home
- He had an accident
- He likes technology
- Wheelchair equipped with remote access for:
  - Lights and shutters (KNX)
  - Multimedia (UPnP)

Their needs

Both

Medical/technical staff should be able to

- Check their health state
- Check home configuration (shutters, lights, heaters…)

Mrs. Dupont

Some daily tasks should be automated (motion troubles) or reminded (memory loss).

Mr. Doe

Would like to control everything remotely, with a unified protocol

Different variability dimensions

- Protocols
  - Low-level protocols: KNX, X2D, X10, etc
  - High-level protocols: UPnP, DPWS, etc
- Devices
  - Lights, heaters, shutters, etc
- Languages
  - Mainly French
  - But also main European languages
- Adaptation to Handicap
  - Motion, memory, perception, etc

Challenges

- Explosion of the number of possible configurations
  - 10^{14} possible configurations! → 10^{28} transitions!
- Dynamic Adaptation
  - Evolution of the handicap
  - Houses should be configured remotely
    - No wires to connect/disconnect in the walls
  - No service interruption
    - Rebooting the system cannot be a solution (lives depend on the system)
- Reliability
  - **Safe migration path**
    from a valid configuration **to another valid configuration**
  - Performance issue (time) not critical

Related works

Reliability, Validation

K. Czarnecki et al.
GPCE'06

F. Fleurey et al.
Models@Run.Time'08

J. Whittle et al.
MODELS'07

B. Cheng et al.
ICSE'06, AOSD'09

∩

E. Figueiredo et al.
ICSE'08

Oreizy et al.
ICSE'98

OSGi, Fractal, OpenCOM, etc

Variability Management

Dynamic Adaptation

Hallsteinsen et al.
Computer'08

Garlan et al.
Computer'04

S. Appel et al.
ICSE'06

M. Mezini et al.
FSE'04

B. Morin et al.
MODELS'08

P. David et al.
SC'06

Validation *VS* Variability management…

- How to validate DAS?
  - Specify everything!
    - all the configurations: $>10^{14}$
    - all the transitions: $\sim 10^{28}$
  - Model checking, code generation

- Problems
  - Explosion: Time consuming, error-prone
  - Evolution of the system (not predicted)
    - Stop all -> Evolve the specifications -> model check -> re-generate -> re-deploy

Validation *VS* **Variability management**…

- How to manage dynamic variability?
  - Do not focus on configurations!
    - Write reconfiguration scripts, encapsulating « features »
  - Depending on the context and/or user needs
    - Choose the most adapted scripts
    - Executes all the selected scripts to dynamically adapt the system

- Problems
  - Scripts written by hand (calls to reconfiguration API)
  - Interactions, dependencies between scripts?
  - Does the configuration (after executing scripts) make sense?
    - Hopefully yes…

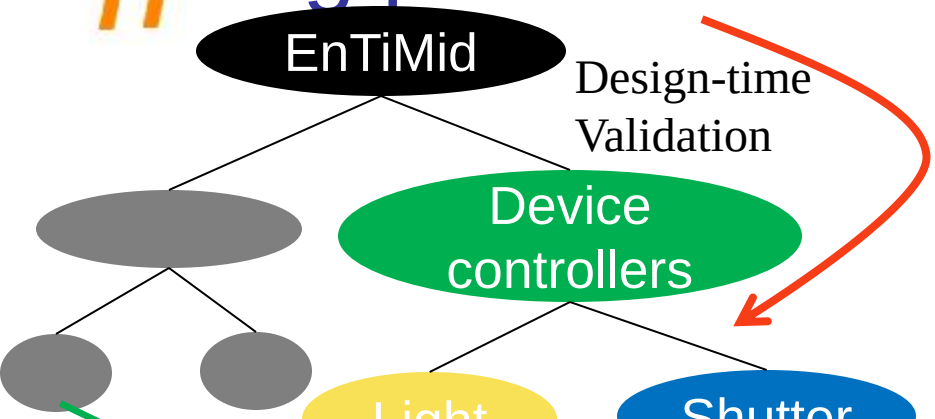
Outline

- Context, Problems and Related Work
- Solutions to meet the challenges
- Conclusion
- Future works

Adopting a DSPL approach

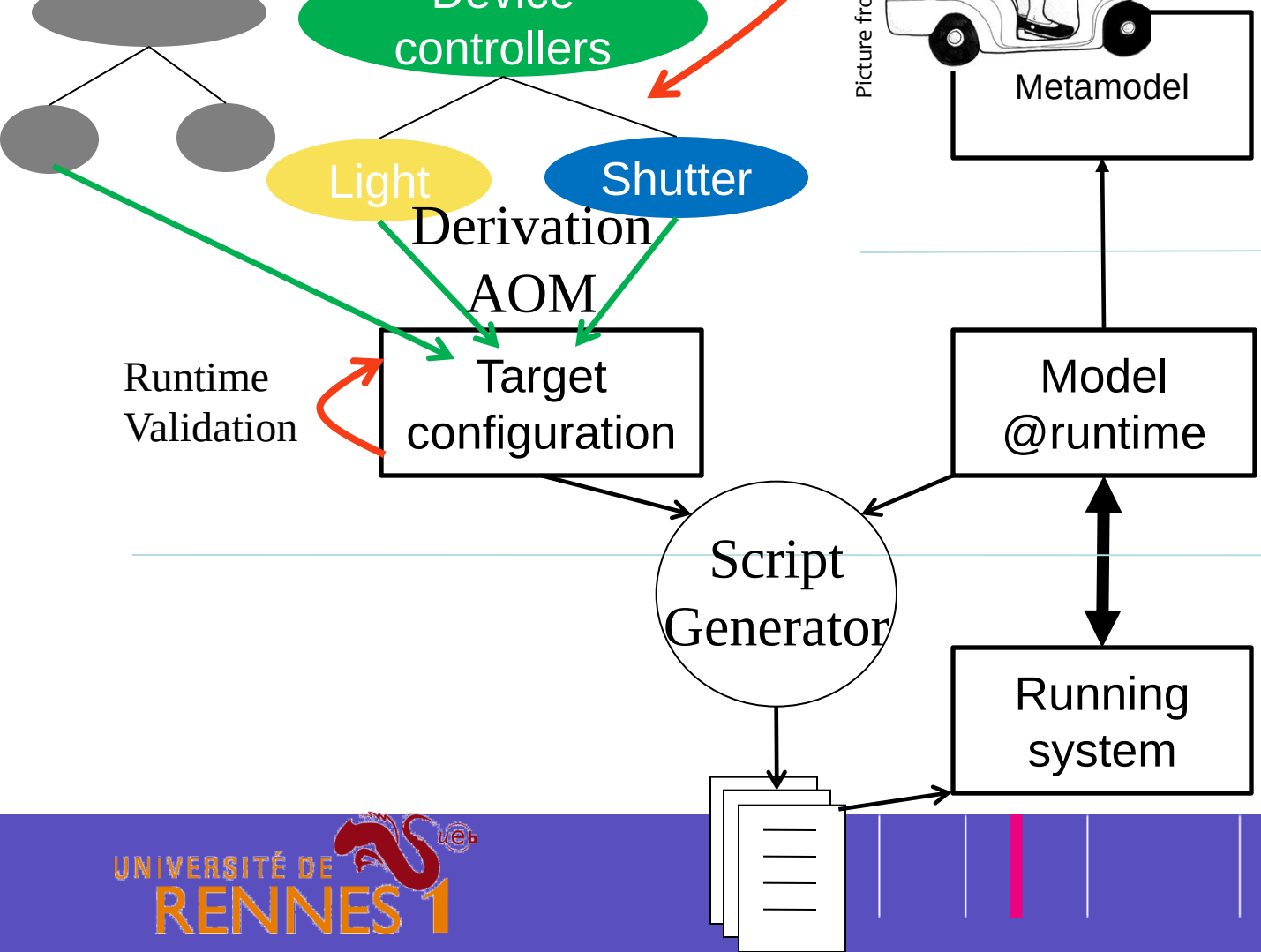
- Focus on variability, not on configurations
- Build (derive) configurations when needed
- Validate configurations before actual adaptation
- Automate the reconfiguration process

Big picture



Had a flu last week
→ stayed in bed
→ a nurse every day

Now in a better shape



Extensive design-time validation

- Still possible to validate everything, for small systems
  - Produce all the possible configurations by aspect weaving
  - Validate all the configurations

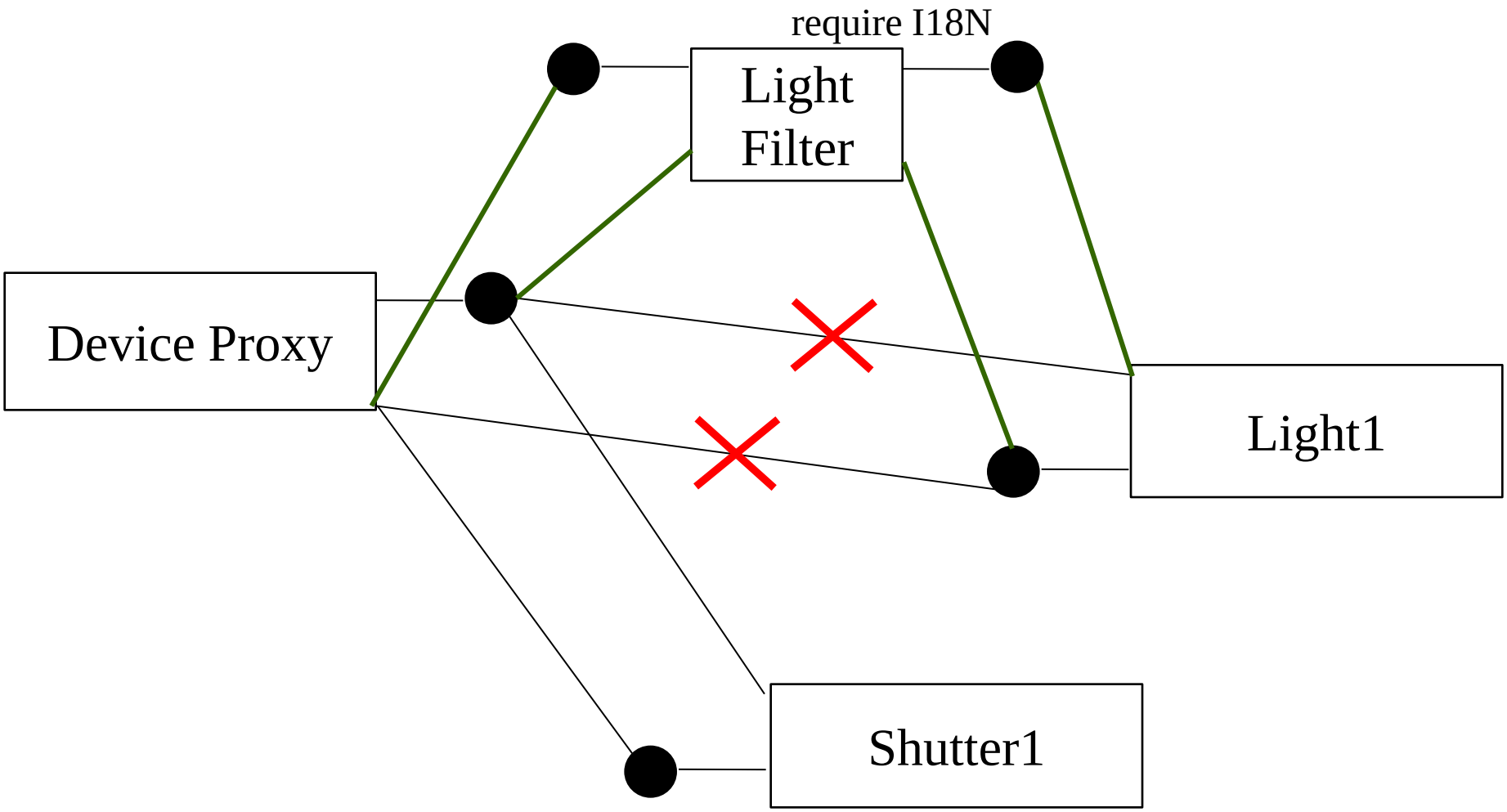
- Discussion
  - Time/resource consuming
  - The number of configurations explodes
  - … but they are automatically generated, by aspect composition

- Not scalable

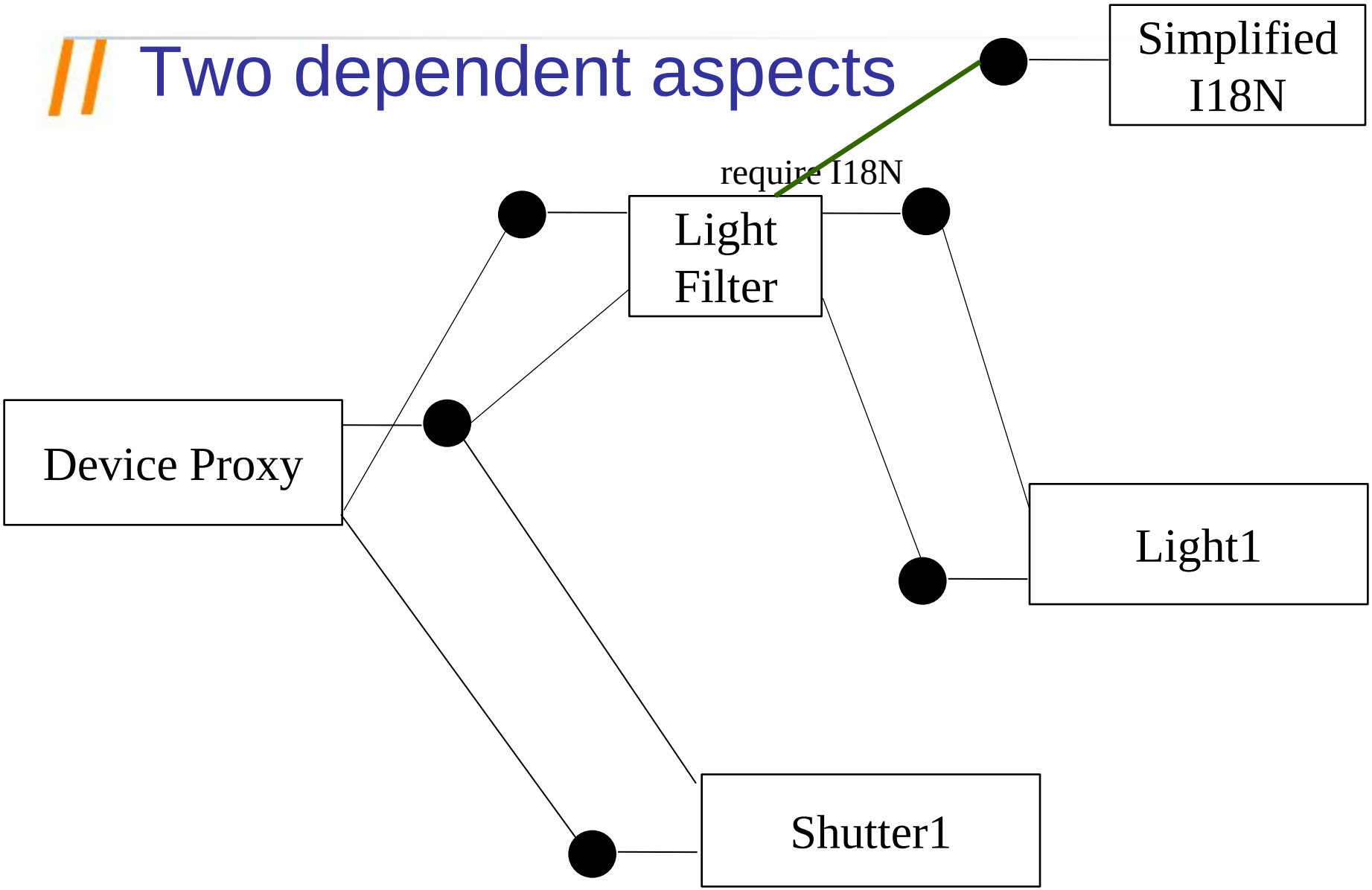
Validation of aspect models

- Aspect-Oriented Modeling
  - Validate the DSPL at design-time
  - Strong theoretical background (graph theory)
  - Modular reasoning
  - → interactions and dependencies detection
    - Using Critical Pair Analysis
  - → weaving order

Two interacting aspects



Two dependent aspects



Limitations of CPA

- Critical Pair Analysis has limitations
  - Aspect1, Aspect2 → OK
  - Aspect1, Aspect3 → OK
  - Aspect1, (Aspect2, Aspect 3) → ?

- Need to validate woven configurations
  - At runtime, when they are produced

Checking configurations at runtime

- Focus on one configuration
  - Not the whole dynamically adaptive system
- Efficient roll-back
  - The running system is not yet adapted
  - Just discard invalid models
  - Report to user

Invariant checking

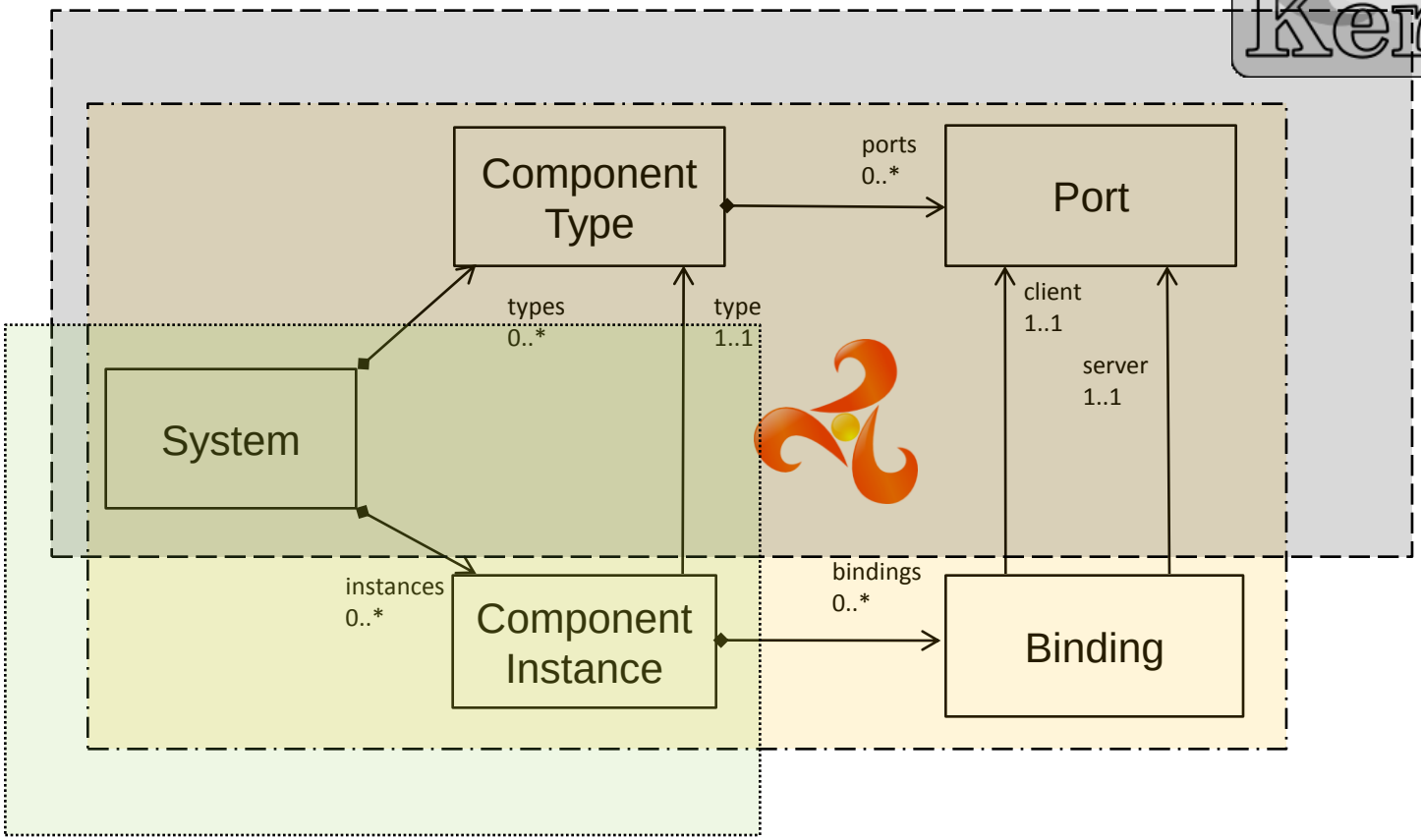
General Invariants

```
aspect class Component {
  inv mandatoryClientPortBound is
  do
    self.type.ports.select{p |
      not p.isOptional and
      p.role == PortRole.CLIENT
     }.forAll{p |
         self.binding.exists{b |
             b.client == p
         }
     }
  end
}
aspect class TransmissionBinding {
  inv wellFormedBinding is
  do
  //link a client port to a server
  //port of the same type
  end
}
```

Specific Invariants

```
aspect class System {
  inv hasEnglishI18N is
  do
    self.allComponents.contains{c |
      c.type.services.contains{s |
          s.name == "org.entimid.I18N"
      } and c.name == "EN"
    }
  end
}
```

Defining checking strategies



Simplified Metamodel

Outline

- Context, Problems and Related Work
- Solutions to meet the challenges
- Conclusion
- Future works

Conclusion

Explosion of the number of possible configurations

- DSPL to manage variability
- AOM to automatically derive configuration

Dynamic Adaptation

- Reflection model causally connected to the running system
- Changes not directly reflected

Reliability

- At design-time
  - Still possible to validate all the possible configurations
  - AOM provides more scalable mechanisms
- At runtime
  - focus on one configuration
  - Efficient roll-back

Perspectives and on-going works

Dual-view AOM

- Structural + behavioral view
- More advanced validation (deadlocks, livelocks, invariants)
- Simulation (performance, impact on QoS)

Towards higher-level adaptations

- We still manipulate components and bindings
- →drive the adaptations using domain concepts:
  device, scenarios
- Use MDE to map domain concepts to architecture

Thank you

Questions?