Construction d'applications dans la ungle de l'IOT

DIVERSE : DIVERsity-centric Software Engineering

HEADS

Software intensive systems

THALES

DGA

AREVA

EDF

software AG

M2M zone

T Jungle

Heterogeneous and Distributed Services

- **Heterogeneous:** The infrastructure on which the service runs is composed of a set of different nodes and networks
  - The “IOT Jungle" which ranges from microcontroller based sensors and devices to cloud.
- **Distributed:** The implementation of the services is composed of a set of independent processes communicating asynchronously.
  - Truly distributed services implementation is required in order t provide useful and reliable services which take advantage of the infrastructure.

xamples

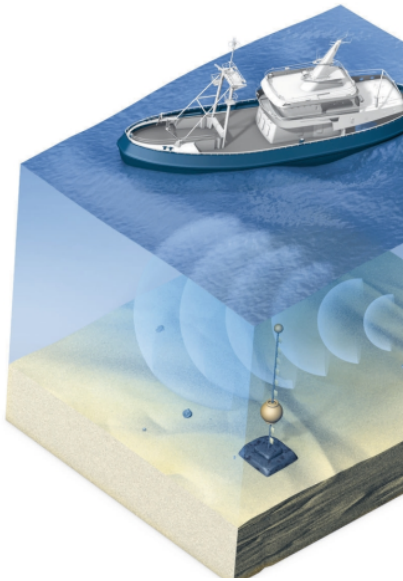
Health domain and ambient assisted living

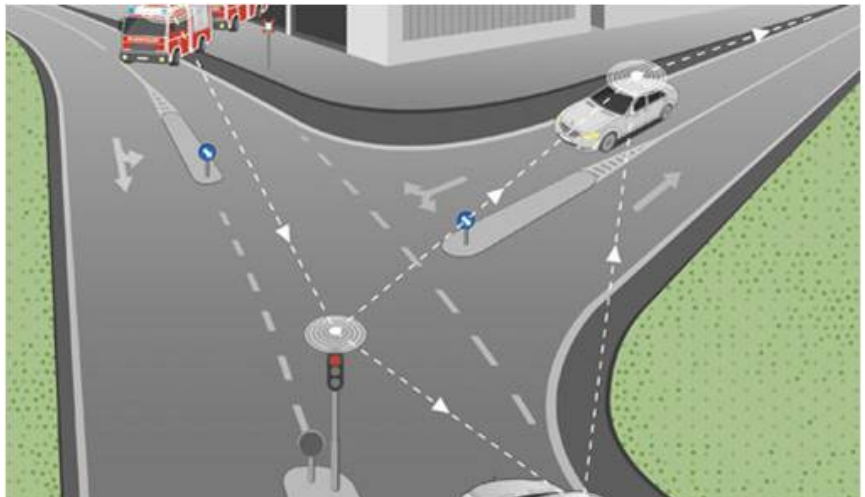
Energy domain and smart grids

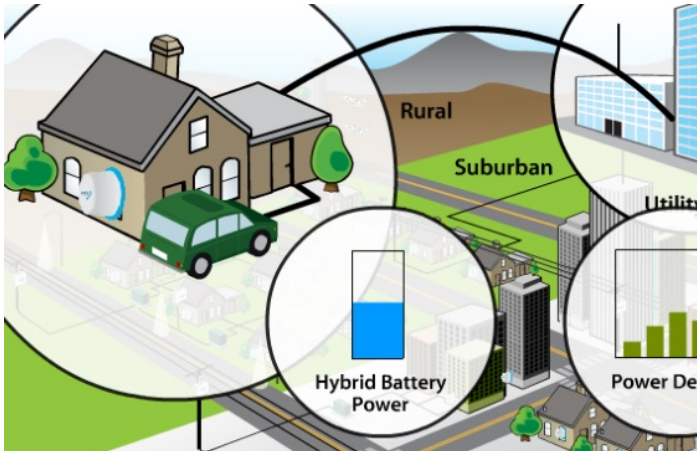
Environmental monitoring and oil and gas

Safety in hazardous environments

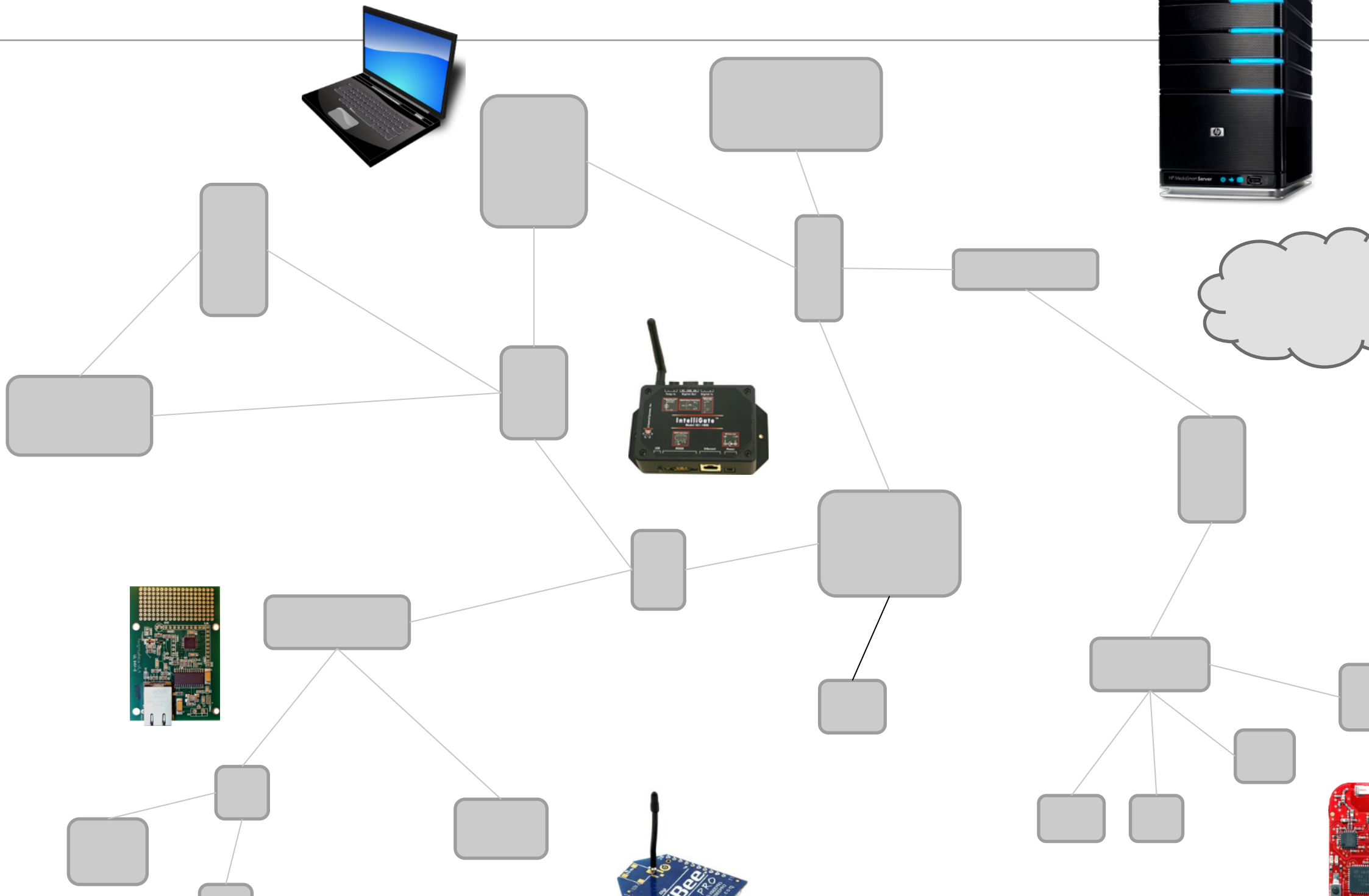
Intelligent Transport Systems (ITS)



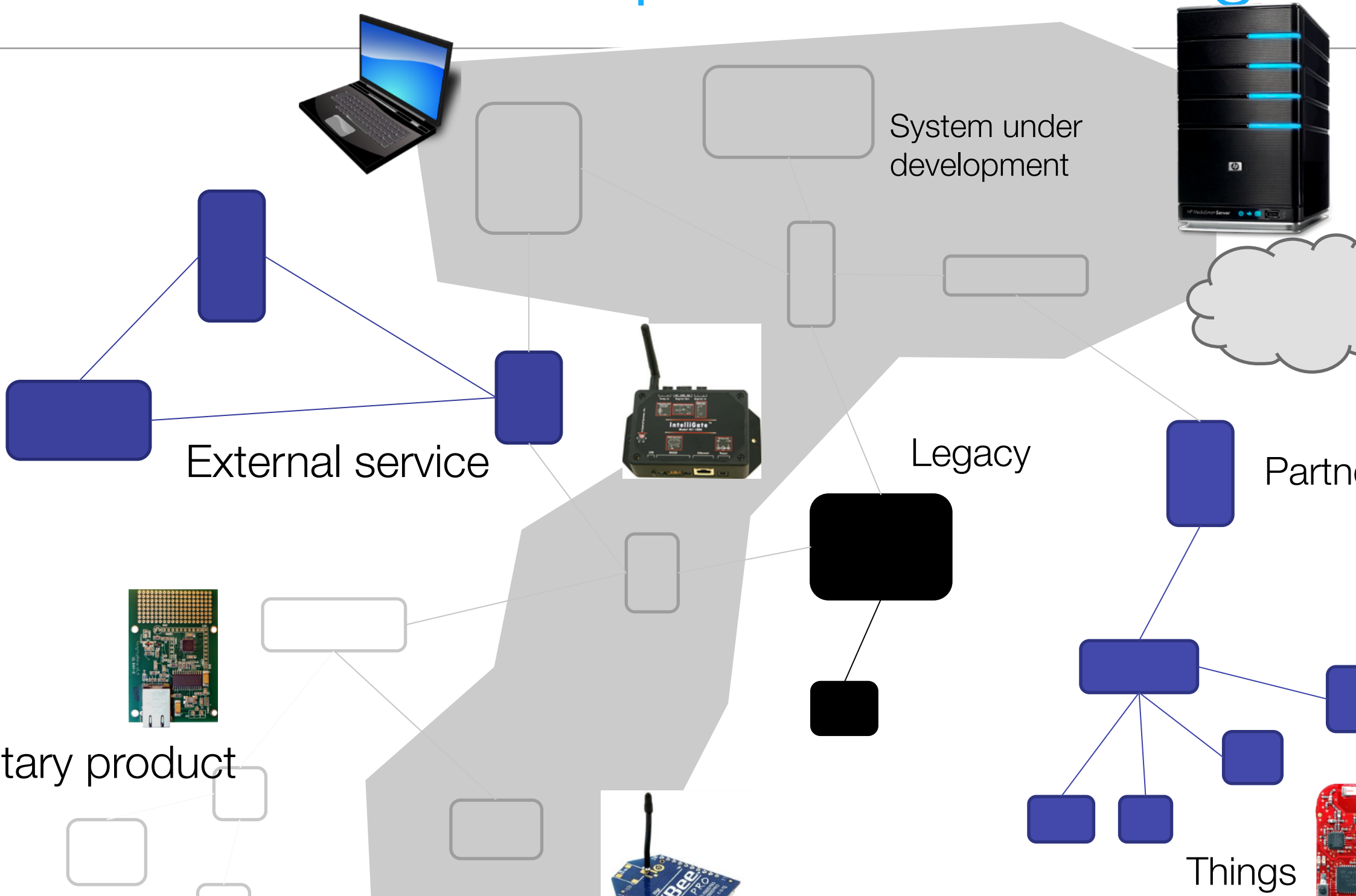




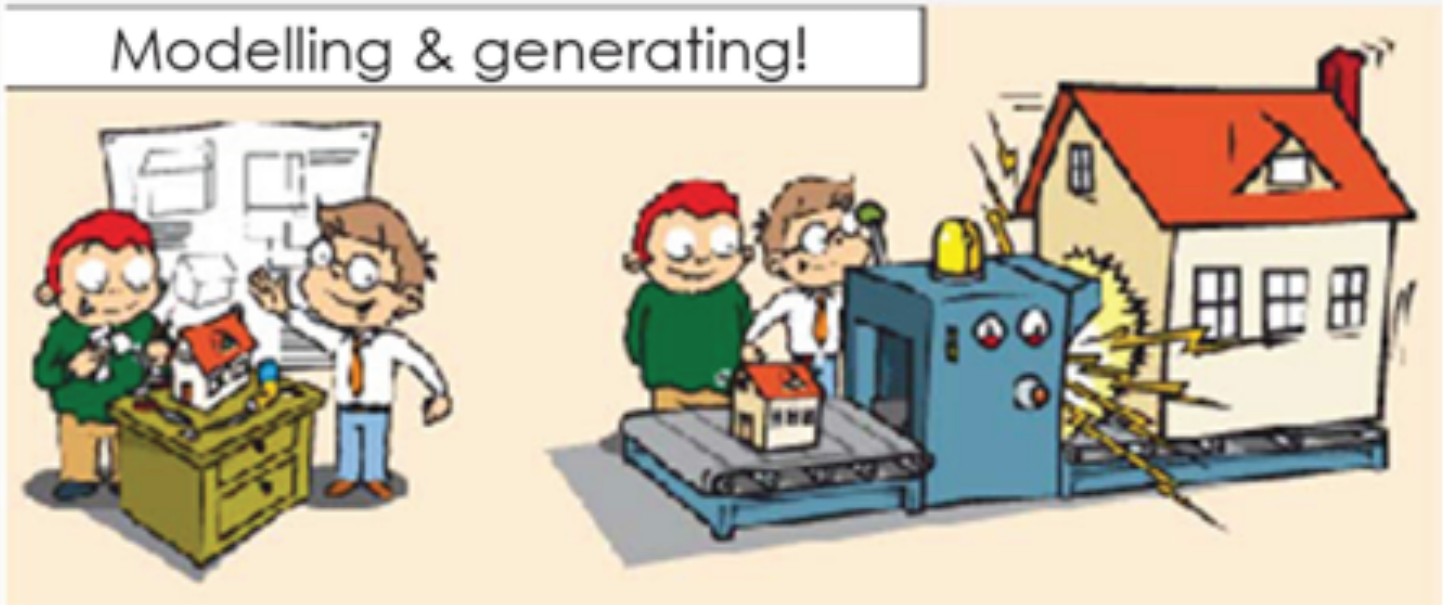
eterogeneous infrastructure



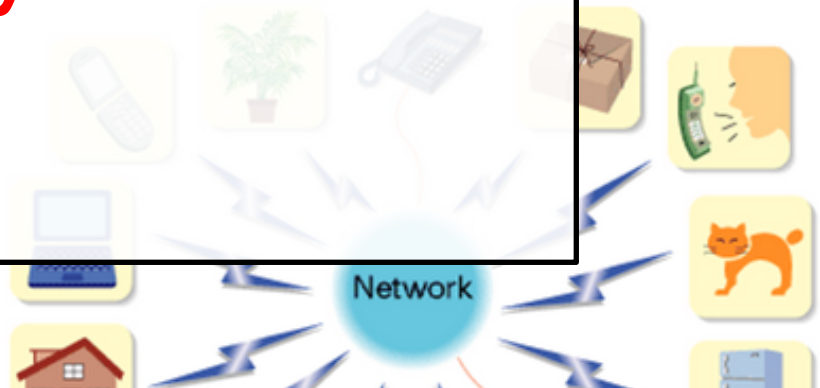
ew Services on top of the IoT Jungle



	$3-$30	~$35	$40	$8	~$3
	Shield ($ ESP8266				Built-in
ming	C/C++				C++/Lua
on	USB/ SPI,				Serial/ OTA
	Built-In				Built-In
	Arduino 13 GPIO/			.6	10 GPIO/ 1 AD
tion	12V 50mA Deep sle		**port**: 346mW **Idle** 88mW **powered** off 45mW	consumption can be as low as 9 µA. 3,3V	3,3V Transmit 802. 11Mbps, POU +18.5dBm 19 Standby 0.9m Deep sleep



Model-driven software engineering to handle diversity



ur Main Objective

Enable efficient exploitation of the broad diversity of the future computing continuum fo rapid service innovation of advanced services by **providing service developers with new agil ool supported software engineering methods enabling short innovation cycles**

Heads project outputs

Two new modelling languages to tame the IoT jungle

What is ThingML ?

A DSL to model distributed reactive systems

- IoT systems, embedded systems, sensor networks, ...

Components, State machines and action language

- « Main stream » MDE

Contributions of ThingML

- « Complete » action language
- Slots, Mixins and Aspects instead of Inheritance and Composites
- Enforced encapsulation and actors semantics

Target Platforms and Applications

- MDE for resource constrained systems (microcontrollers, IoT)
- Development of applications distributed across heterogeneous hardware

Why ThingML ?

Typical MDE benefits

- Reduce development, maintenance and evolution costs
- Perform verifications and analysis on the models
- Model application at a platform independent level

No existing approach can deal with microcontrollers

- ThingML can run on hardware less than 1ko of RAM

No existing approach is really platform independent

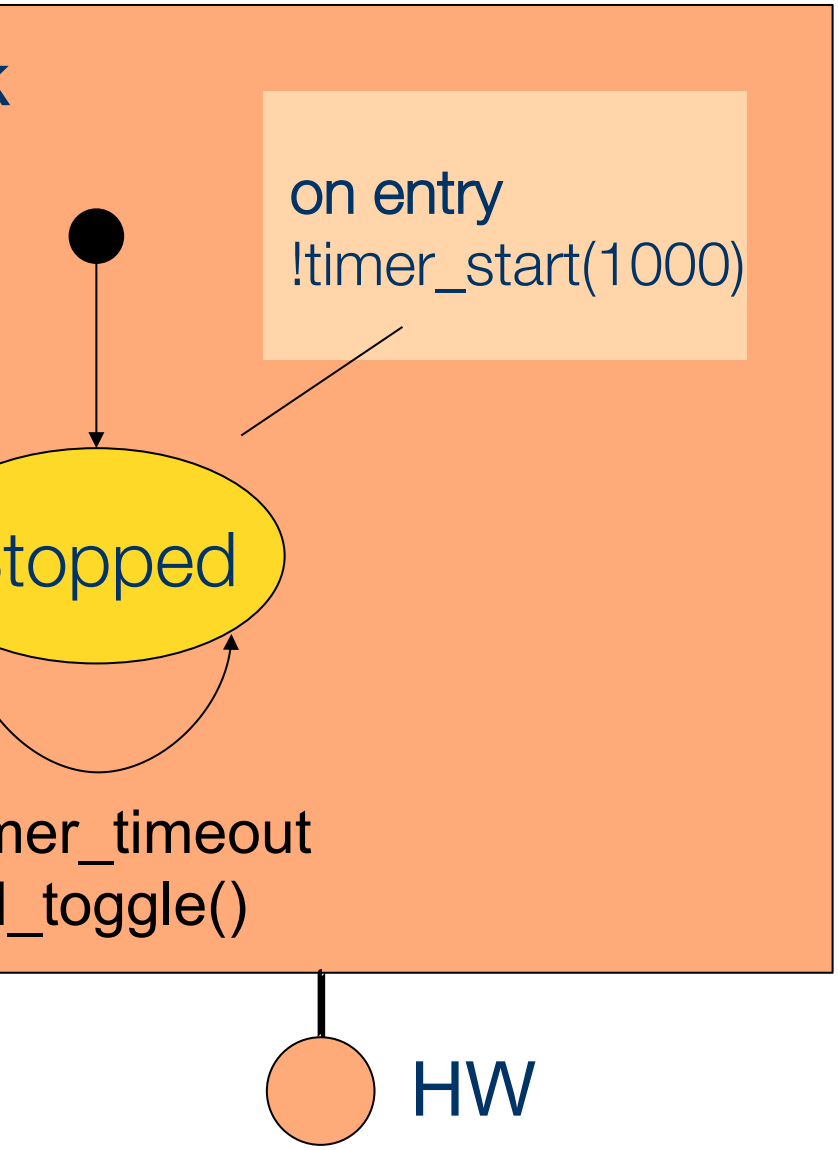
- Since actions are written in the target language

ningML Goals

Provide tools and methods

- For each actor to concentrate on his task
- For decoupling the tasks of different actors
- Using state of the art software engineering practices
  - Modularity, reusability, runtime deployment, continuous integration, validation, etc…
- Cost efficient and practically usable
  - No large overhead, integrated with legacy systems, etc…

Blink example state machine



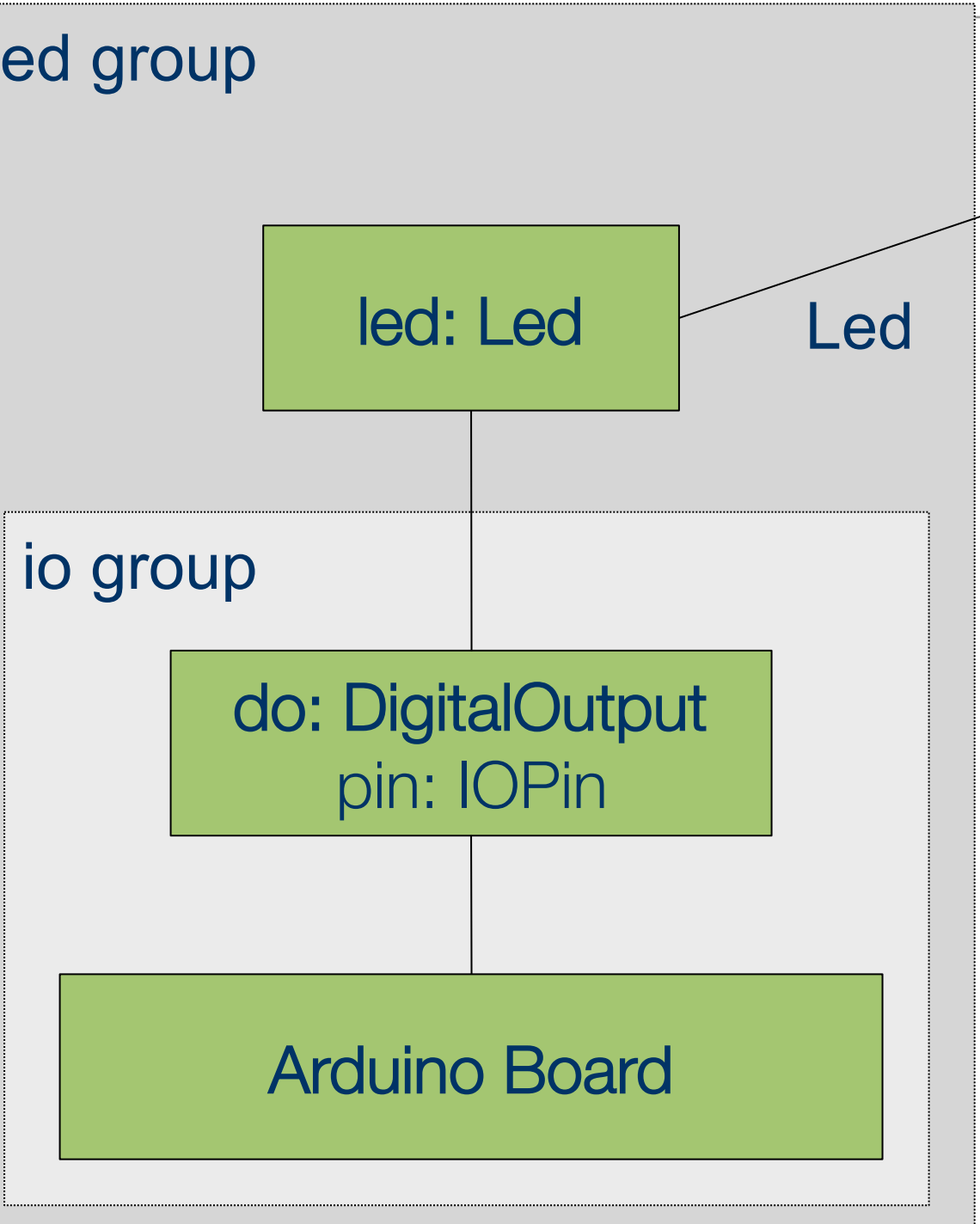
= timer_timeout()
> led_toggle()

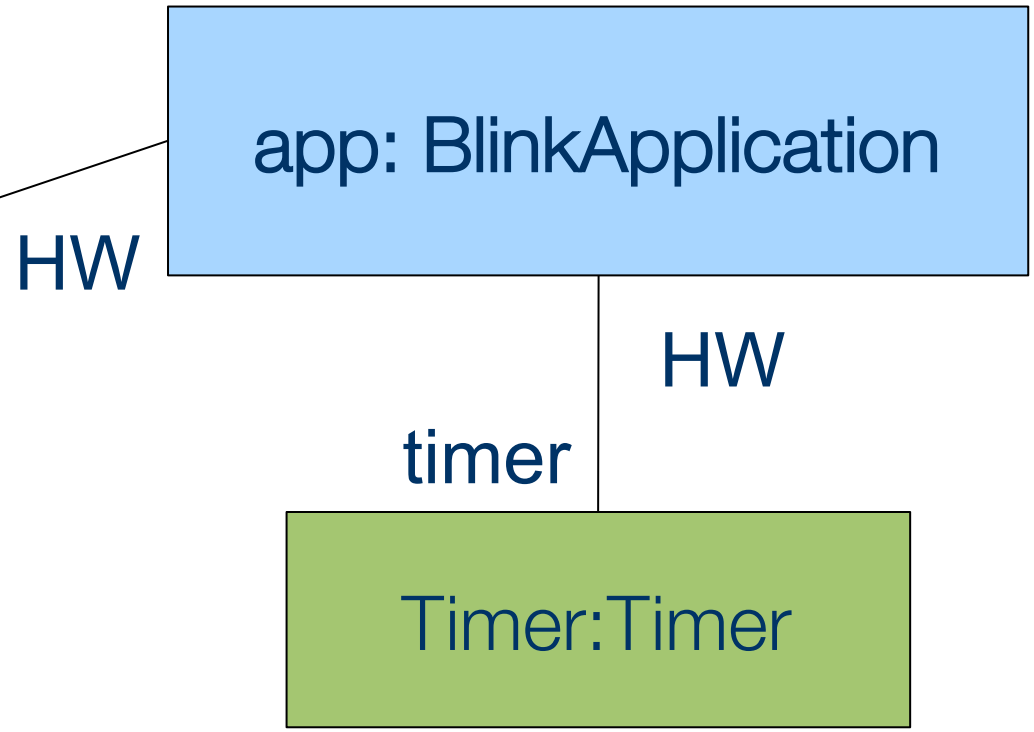
```
thing Blink includes LedMsgs, TimerMsgs
{
    required port HW
    {
        sends led_toggle, timer_start
        receives timer_timeout
    }

    statechart BlinkImpl init Blinking
    {
        state Blinking
        {
            on entry HW!timer_start (100

            transition -> Blinking
            event HW?timer_timeout
            action HW!led_toggle ()
        }
```

link example and instance groups






```
configuration BlinkArduino
{
group led : LedArduino
set led.io.digital_output.pin = DigitalPi
// The timer
instance timer : TimerArduino
// The blink application
instance app : Blink
connector app.HW => led.led.Led
connector app.HW => timer.timer
```

The ThingML tools

Based on Eclipse / EMF Metamodel

Textual Syntax with EMFText

- For good usability and productivity
- To keep the development cost of the editor(s) reasonabl

Graphical exports (graphML, graphviz, …)

Static well formedness and type checker

Equivalent compilers for a set of platforms

- C/C++ for different microcontrollers, linux, embedded lin
- Java for computers, smartphones, …
- Javascript (NodeJS)
- Maybe others if needed

Generators for communication channels

Easy to distribute ThingML IDE

hingML Editor

ThingML Editor : blink.thingml

```
4  thing Blink includes LedMsgs, TimerMsgs
5  {
6      required port HW
7      {
8          sends led_toggle, ti
9          receives timer_timeo
10     }
11
12     statechart BlinkImpl ini
13     {
14         state Blinking
15         {
16             on entry HW!time
17
18             transition -> Bl
19             event HW?timer_t
20             action HW!led_to
21         }
22     }
23 }
24
25 configuration Blink
26 {
27     group led : Led
28         set led.io.digital_o
29     // The timer
30     instance timer : SoftTim
```

Compile to Arduino | Compile to S

Blink | Arduino 0022

File Edit Sketch Tools Help

Serial Monitor

Blink

```
/*********************************************************
 * File generated from ThingML (Do not edit this file)
 *********************************************************


/*********************************************************
 * Definition of simple types and enumerations
 *********************************************************

// Definition of Enumeration  PinMode
#define PINMODE_INPUT 0
#define PINMODE_OUTPUT 1

// Definition of Enumeration  DigitalState
#define DIGITALSTATE_LOW 0
```

Done uploading.

```
Binary sketch size: 3490 bytes (of a 258048 byte
maximum)
```

1

onfiguration and deployment

Heterogeneous execution platforms

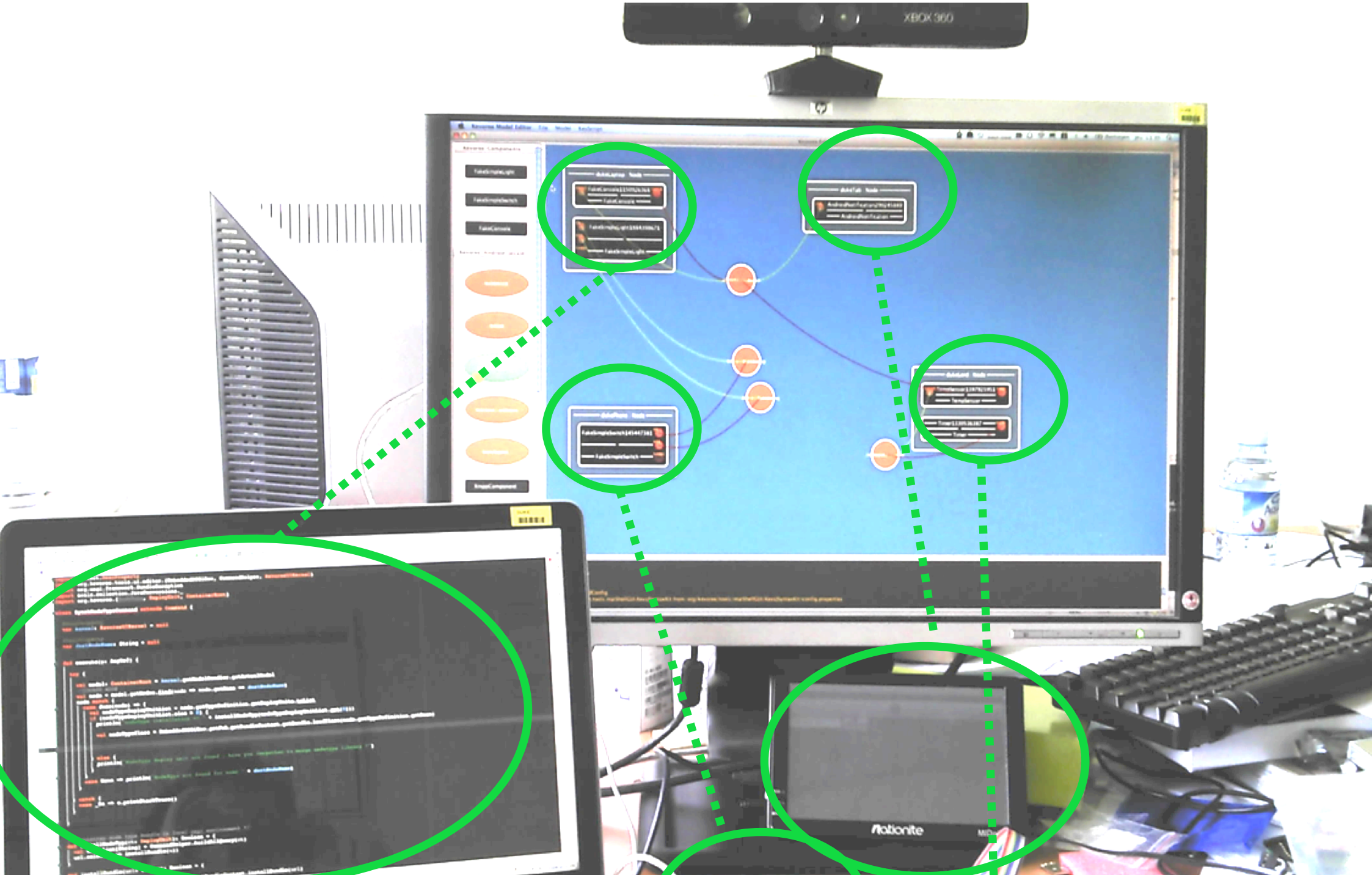
- how to handle deployment?
- how to handle reconfiguration?
- how to verify system level properties?

Effective support for runtime models

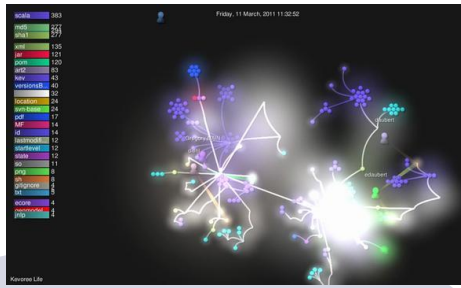
- limited resources
- time constraints

Distributed model dissemination

eterogeneous systems



Kevoree in nutshell



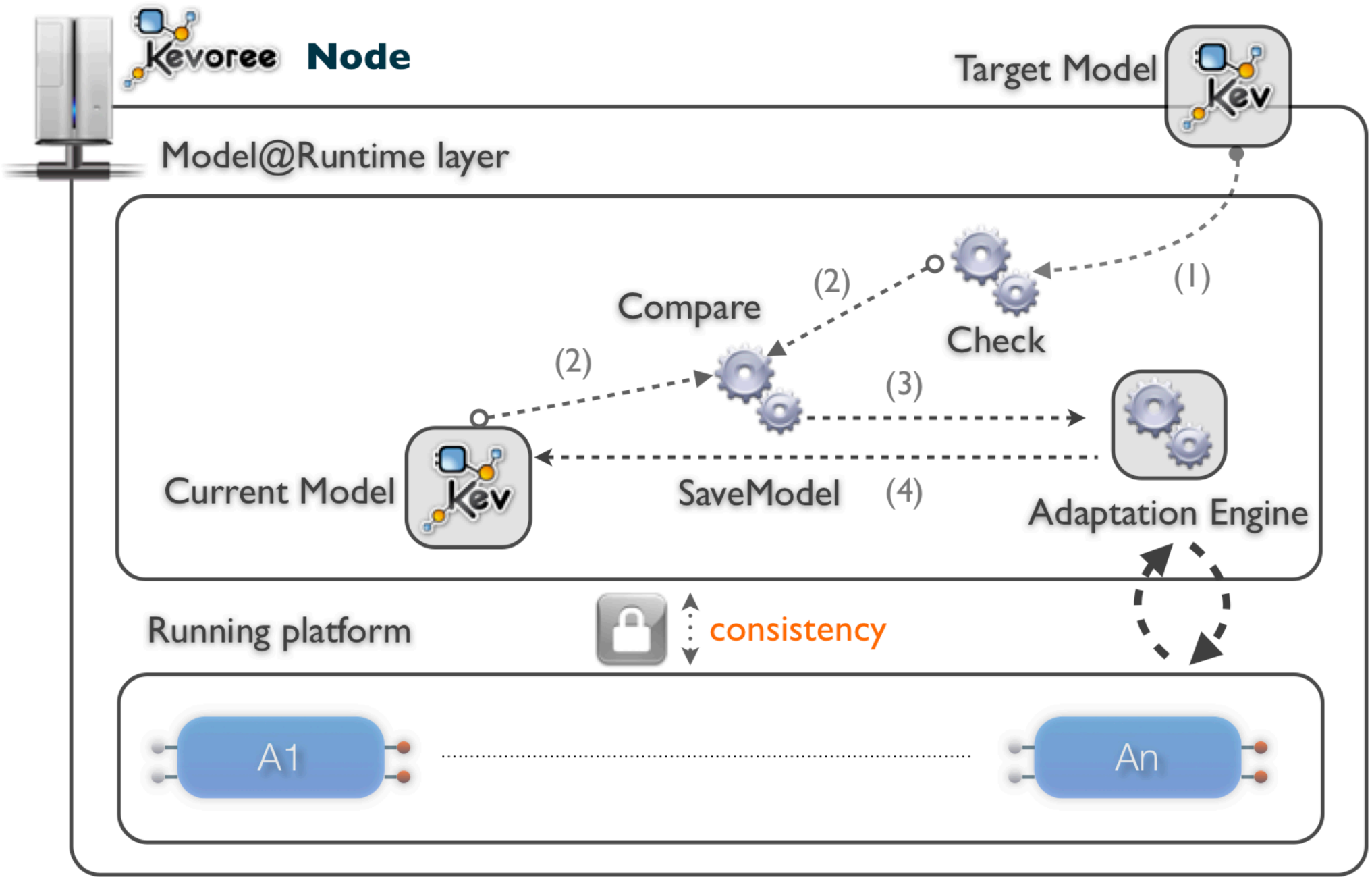
Resilient Software infrastructure based on diversity

Extensible Virtual System Infrastructure

M@R Runtimes for Distributed and Heterogeneous adaptive systems

Lightweight M@R framwork for building Distributed Adaptive

Kevoree in nutshell



Kevoree in nutshell

kevoree.org

MesNotes | iGoogle | (Fractal API) | Jacquard | http://cisedu.u... | [pcinpact.com]... | ADE Production | session-proper... | Da Linux Frenc... | →Anglais | Bienvenue sur ... | KerMeta — Ke... | Hotmail | Lexibase

KEVOREE PROJECT | Home | Documentation | Download | Tutorials | KMF | Polymer | About | News Archives

Kevoree

WHAT IS KEVOREE?

Kevoree project aims at enabling distributed reconfigurable software development. Build around a component model, Kevoree leverages model@runtime approach to offer tools to build, adapt and synchronize distributed systems.

WHY KEVOREE?

Extensible, this project already offers runtime for JVM, Android, Arduino but also for virtualization management such as VirtualBox. In short Kevoree helping you to develop your adaptable software from Cloud stack to embedded devices !

MAKE IT EASY

Kevoree offers an abstraction model that makes easier the manipulation of concepts such as Nodes, Components, Synchronization groups and Communication channels in distributed software systems.

DISTRIBUTED MODELING

Modeling explicits the nodes and the topology of distributed systems. As a system representation, Kevoree allows the model itself to be distributed across the nodes to share the cluster oraganization.

EVENT DRIVEN, SCALABLE

Build around asynchronous model, Kevoree allows design of asynchronous communication, asynchronous synchronization.

OPEN SOURCE

Licensed under LGPL and maintained by the community on GitHub.

CONTINUOUSLY DESIGN

Because long living system must evolve while serving, our approach takes care about HotSwap functionality.

BUILD YOUR OWN CLOUD

Particular software always need particular optimization. Embeddeding your own adaptation layer in Kevoree cluster allows you to build your own cloud.

NEWS

Kevoree 3.0.5 out ! — DECEMBER 7, 2013

Two news, first of all the first version of Kevoree 3.X.X branch is out with version number 3.0.5 and respectively 3.0.2 for base library. In addition we will be present in China @Middleware conference to present it. See you at Beijing !

KMF 1.7.0 — OCTOBER 13, 2013

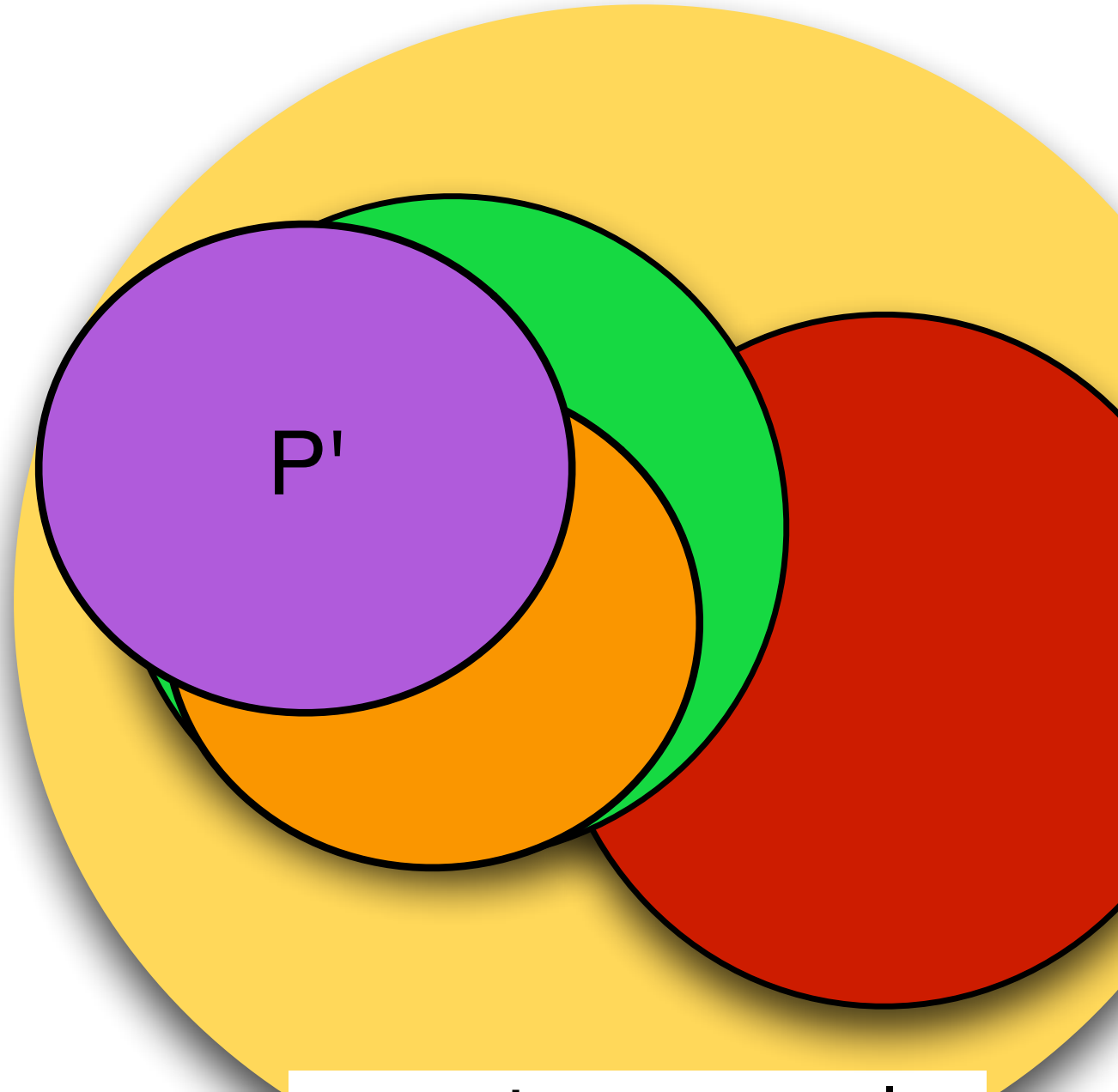
KMF 1.7.0 is now released ! With many new features for aspects oriented devepement and better scalability for JS platform. Two new samples are coming to explain how to build an ecore web editor with KMF … stay tuned … Duke :-)

Current Works

easonable correctness envelop

Automatic synthesis of software diversity

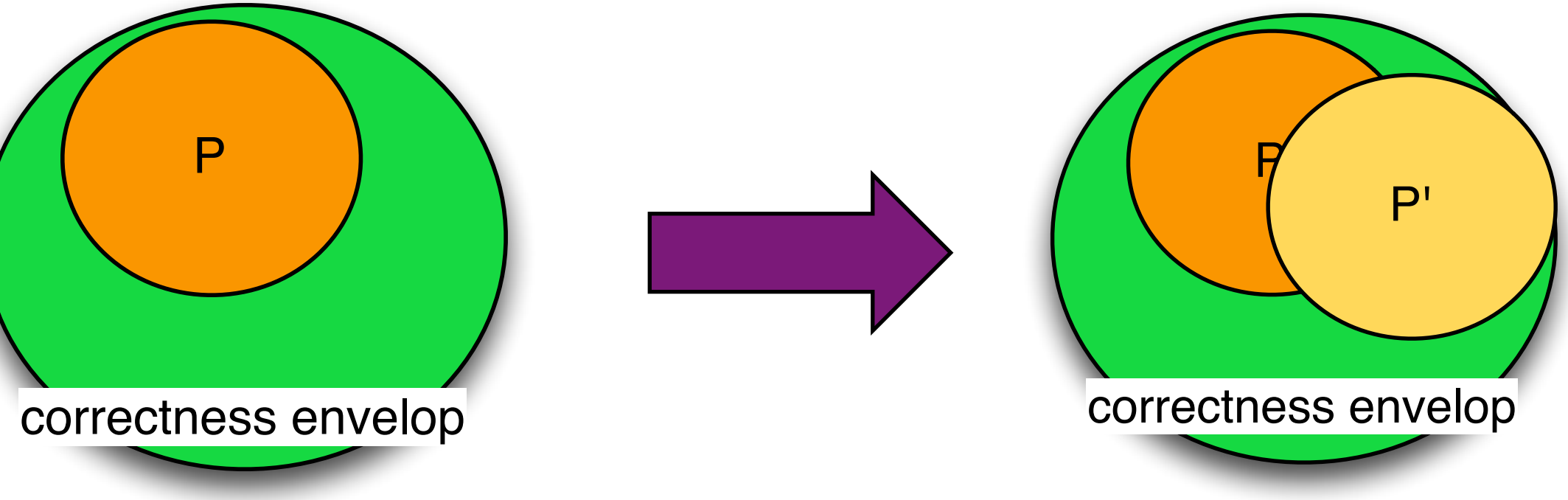
- sosies are different programs that still belong to the correctness envelop



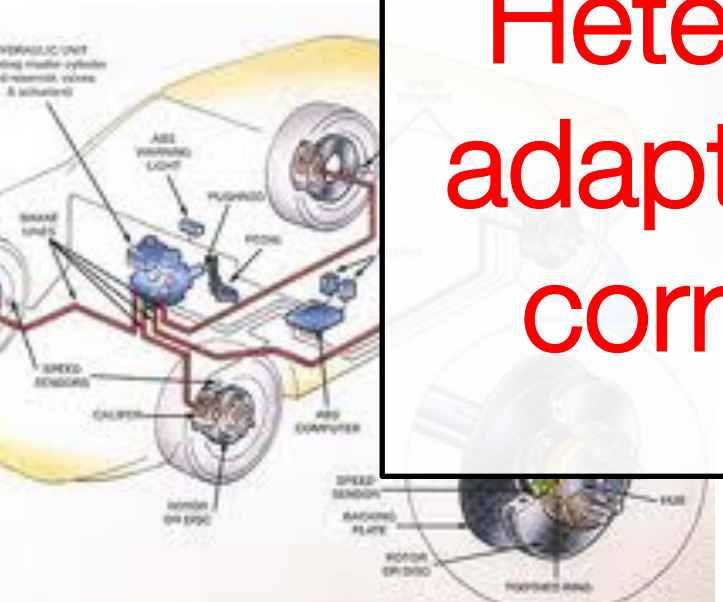
utomatic diversity synthesis

Automatic synthesis of software diversity

- sosies are different programs that still belong to the correctness envelop



oftware intensive systems



Continuous design
for
Heterogeneous, variable,
adaptive within reasonable
correctness boundaries