# Flexibility in embedded real time applications: the Flex-eWare project


Jacques Pulou (Orange Labs , jacques.pulou@orange-ftgroup.com)
Thomas Vergnaud (Thales Communications , Thomas.Vergnaud@thalesgroup.com )

**Consortium of the Flex-eWare project :**

Compagnies : **Thales, France Telecom, Schneider Electric, TEAMLOG/Open , TRIALOG, ST Microelectronics**

Academics and Public Labs : **CEA, INRIA, UPMC/LIP6, Telecom ParisTech**

# Rationale

- More and more embedded and distributed Systems
- More and more complex software systems (context aware, autonomous, open, heterogeneous, always varying,…)
  - Capturing more and more added value…
  - Committed to change rapidly … to maintain existing or new devices on the leading edge of innovation
  - Endless growth of software development cost and delay,
  - Software Quality is still a challenge !
- Component Based (CB) software as a possible solution !
- Flex-eWare Project aims to improve productivity through :
  - Cross fertilization of existing CB technologies
  - Taking into account software “flexibility” requirements

# Component Based Software Engineering (CBSE)

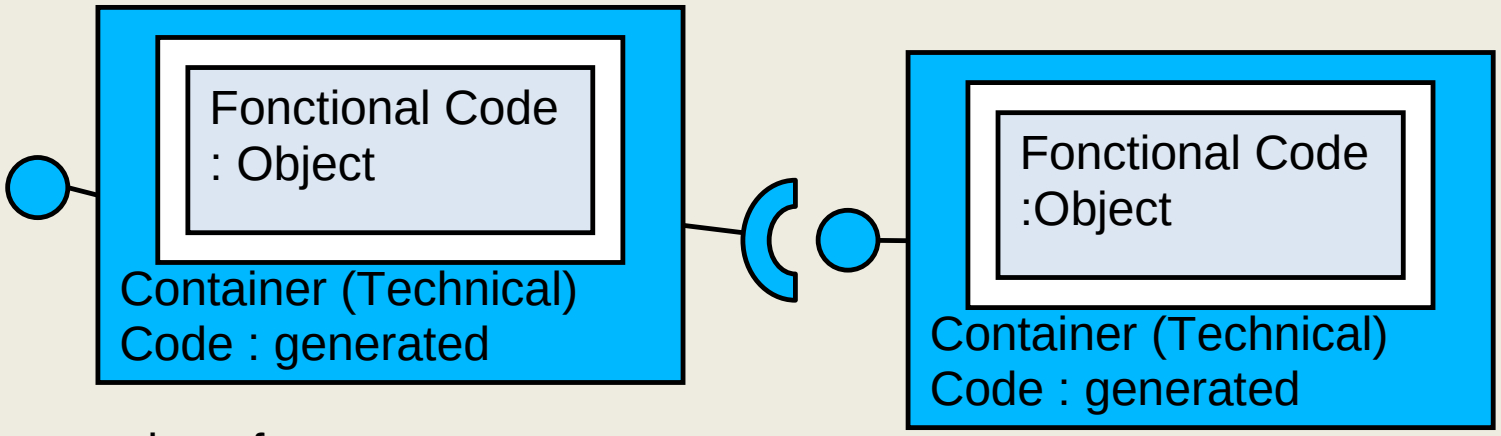
Architectural Model

Component Model and Composition scheme + Component functional source code

Developed “from scratch” or selected in an existing library (**re-use**)

**Separate** functional concern from non functional ones (concerns can be modified **independently** each others)

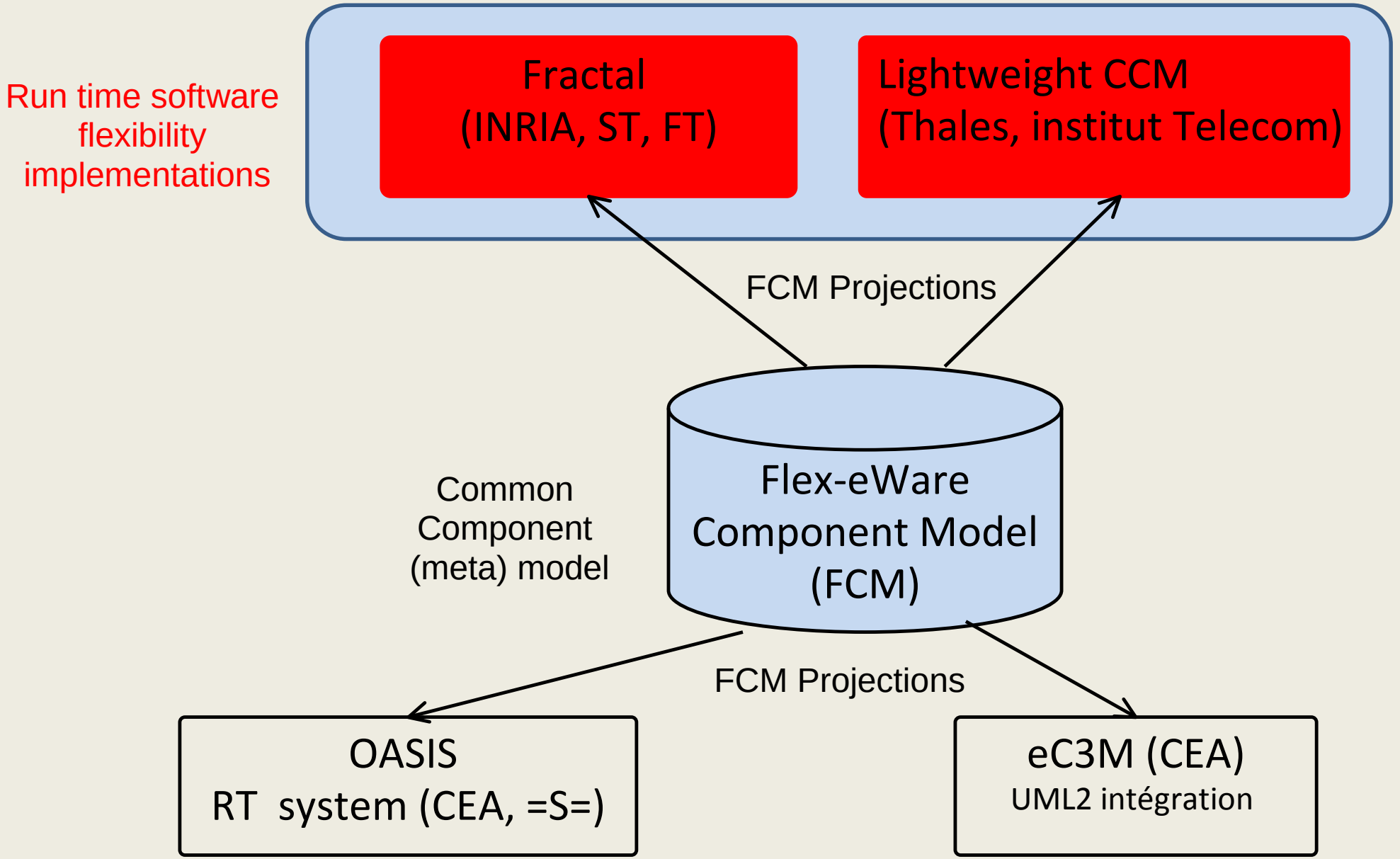
Compiling and Deploying




Explicit programming of dependencies between component exposes the **software architecture**

Applicative codes are integrated by means of **component composition**

# Flex-eWare : cross fertilization and flexibility



# Flex-eWare Component Model (1)

- « Pivot » component and architectural model
  - Common component model unifying partners ones
  - Capture requirements of different application domains from industrial partners :
    - ST Microelectronics, France Telecom, Schneider Electric et Thales

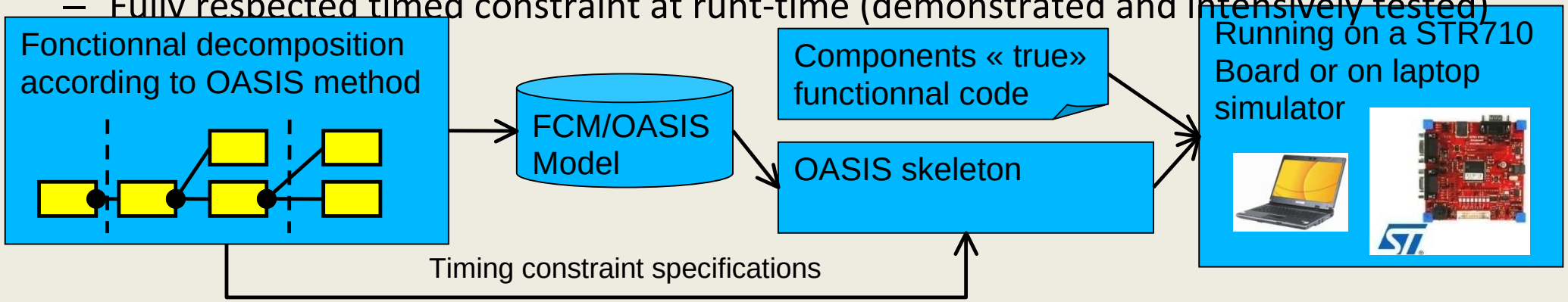
- "Front end" for Available implementation technologies:
  - Lightweight CCM ➔ eC3M or MyCCM
  - Fractal ➔ Cecilia or Think
  - OASIS (programming model and its tool chain)

# Flex-eWare Component Model (2)

- Approach
  - A “core” capturing common features
    - Components (types, implementations, instances), ports, communications abstraction(connectors)
  - Extensions for domain specific requirements
    - communication semantic, deployment, etc.
- Results
  - Meta-Model  FCM that encompasses architectural software definition (component, port and so on) and deployment
  - UML profile  implementation  of the FCM  (compliant with the MARTE profile)

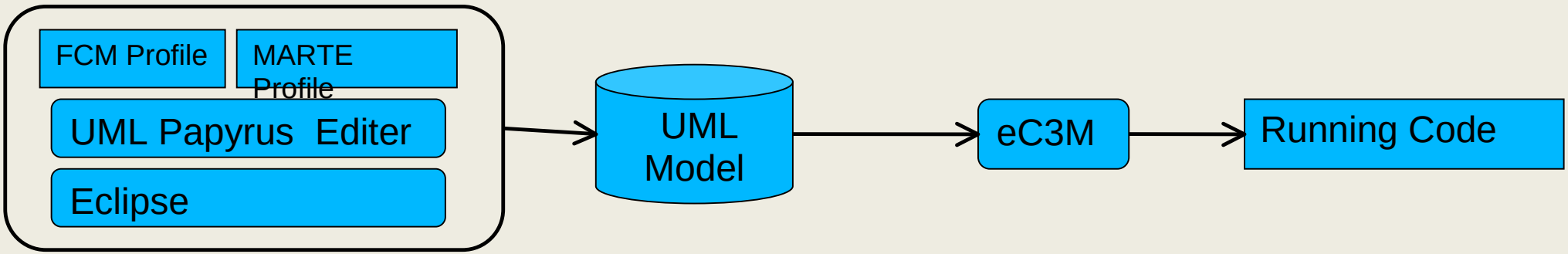
# FCM Integration in the OASIS design flow

- OASIS (CEA) :High integrity and hard real time constraints systems
  - used for nuclear plant control
  - Time triggered static scheduling ( « good by construction »), overrun checked at run time
- Use case : low voltage Schneider Electric digital breaker
  - Maximum delay between default occurrence and circuit opening : less than 26,64 ms
  - STR710 evaluation Board (ARM 7, 256 kBFlash, 64 kBRAM, CanBus, USB, SPI, 4 UART,…)
- FCM integration in the OASIS design flow
  - Demonstrated on an OASIS target including a user interface : STR710 or Simulation
  - OASIS specified timing constraints
  - Fully respected timed constraint at runt-time (demonstrated and intensively tested)



# FCM Integration in UML2 : eC3M

- Tool chain based on the UML FCM & MARTE Profiles
  - Implementation of the FCM profile with Papyrus UML
    - http://www.papyrusuml.org
  - code generator based on the UML FCM & MARTE Profiles
    - http://www.ec3m.net

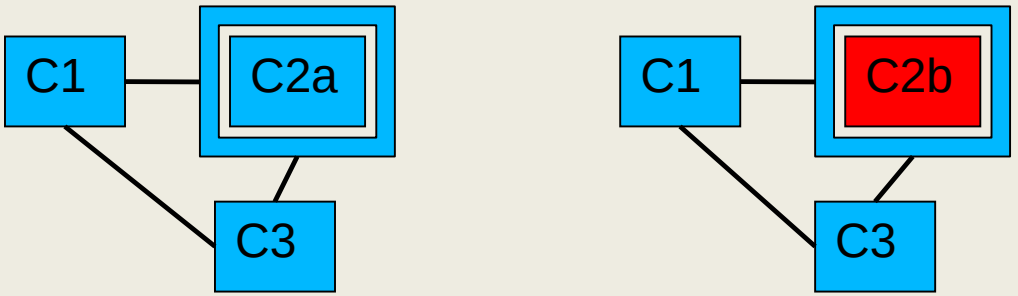


# What is software flexibility ?

- Changing some features doesn't result as a new software !
- Not limited to execution step …see examples below :
  - Modeling and Design step :
    - switching , mixing different models (different execution schemes, legacy,…)
  - Development and Coding step :
    - not limited to some programming languages (DSL possibility,…)
    - separation between functional codes and service (infrastructure) code
  - Configuration and deployment step :
    - choosing between different implementations potentially legacy ones
  - Execution step :
    - replacing some « pieces of code » by another ones eventually legacy ones… (imply some traceability between programming and run-time model)
- Software Flexibility includes :
  - model and code re-usability, dynamic reconfiguration (addressed here )
  - **but also other topics of interest for future work !**

# Dynamic reconfiguration

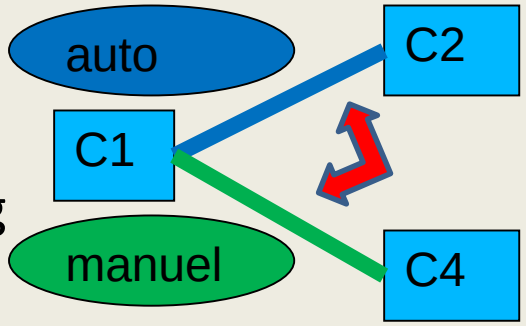
- Opportunistic reconfiguration
  - Tuning the trade-off between « run time architectural capability » and RAM footprint




- Foreseeable reconfiguration : context aware mode switching.
  - Preserving static (i.e. design, development, deployment time) capability of validation and verification

    in spite of context sensitiveness at run time
  - Separate mode functional from mode switching

    mechanism at modeling and coding steps

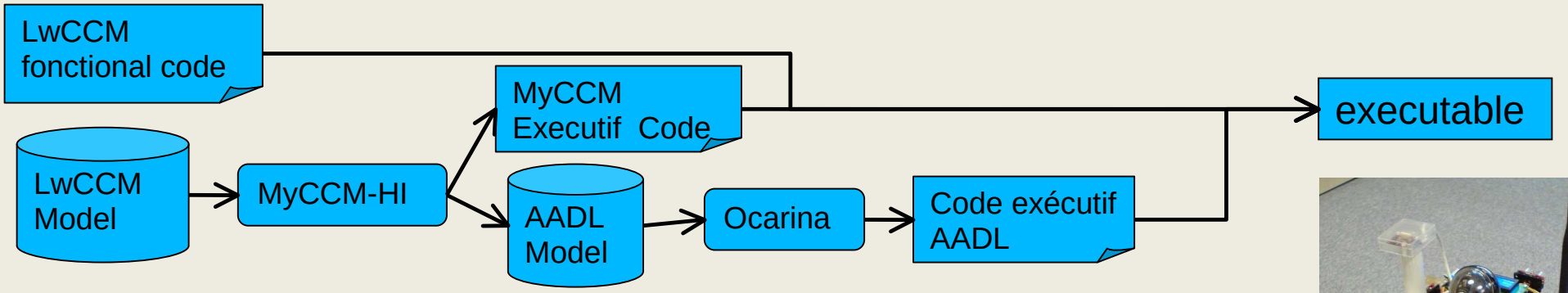


# LwCCM Flexibility (1)

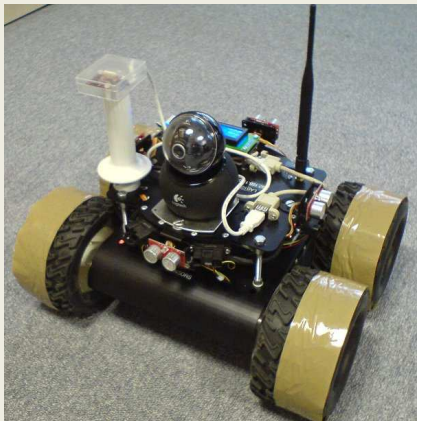
- Lightweight CCM : OMG standard for CB architecture
  - Embedded and simplified version of the CCM standard
  - Application/functional code encapsulation
    - Separate and isolate application code from execution platform
  - component deployment
    - Component composition (Components assembly ing and connecting)
    - Run time resources allocated to entry point (activity)
- Thales specific Implementation : MyCCM
  - http://myccm-hi.sf.net
- Lightweight CCM Flexibility
  - Mode Based reconfiguration of the application
    - pre-determined application configuration and functioning (a.k..a s Modes)
    - Mode switching directly supported by implementation and run time
    - No intrusive for the application code.
  - deterministic, low footprint , real time (deadline guarantee )

# LwCCM Flexibility (2)

- AADL intermediate code : Low level architectural model
  - Architecture description + resource allocation
  - “Schedulability “analysis and memory demand estimation
  - code generation
- Two step generation
  - MyCCM High Integrity (Thales) : application model → AADL
  - Ocarina (Telecom ParisTech) : AADL → executable code




- Semi-autonomous robot as validation use case.
  - switching from manual to automatic control mode



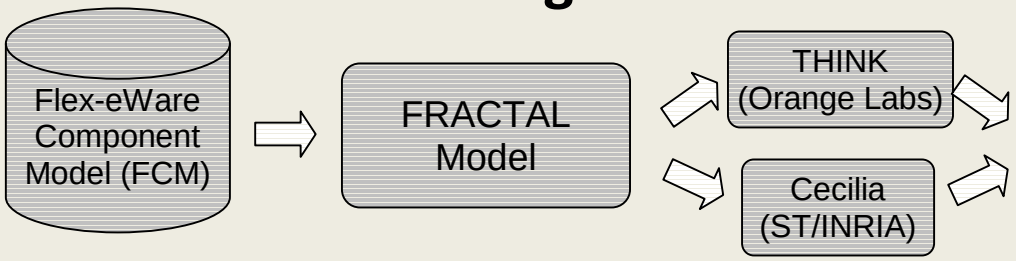
# Fractal Flexibility (1)

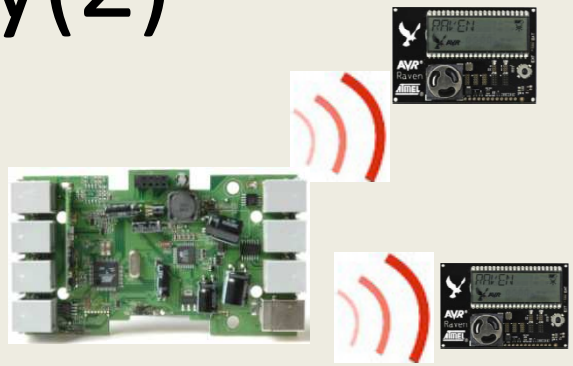
- Fractal : hierarchical and generic component model
  - not a fixed list of extra-functional services, possibly unused (context sensitive)
  - open and versatile container extensible to requirements
  - Several compliance levels regarding services for introspection and intercession
  - http://fractal.ow2.org


- Towards an open source mature Tool Chain
  - Mind Project of the Minalogic Cluster http://mind.ow2.org
- Fractal Flexibility
  - At design level
    - Hardware agnostic (neither to given platforms nor to specific domains)
    - Tuning the trade-off between architecture adaptability / memory footprint
    - Fractal Container adaptation to addressed contexts (eg. concurrency , execution, communication model, resource allocation, etc.)
  - At run time : software upgrade (extension, bug fixing,…).

# Fractal Flexibility(2)

- **Seamless Flex-eWare design flow**



- **Fractal native( C written components, C back end) deployment on tiny target (few KB Ram) with dynamic reconfiguration (up to hot swapping )**
  - architectural context Code optimization : singleton pattern, discarding unused functions or controllers, …. and static references for fixed bounding
- **Generation and automatic deployment of dedicated containers**
  - Concurrent execution resources, distributed hardware target, memory isolation
  - « multicast » Communication modes,…
- **WSN use cases**
  - Deploying an MPEG2 decoder on a distributed platform by means of the Comete middleware ( concurrent threads communicating through TCP sockets )
  - dynamic reconfiguration on the node of a zigbee based WSN.

# Conclusion

- Software engineering contribution
  - Definition of a common component (meta-)model
  - Some flexibility features added to existing approaches.
- ~20 scientific publications
- 5 five tool chains developped or improved :
  - MyCCM-HI,
  - Fractal/THINK, Fractal/Cecilia,
  - OASIS, eC3M)
- 5 phD defended: Polakovic (may 2008), Plsek (sept. 2009), Borde (dec. 2009), Renault (dec. 2009), Poulhiés (mars 2010)

# Future Work

- Dissemination to come
  - Synthesis Paper under submission.
  - ISORC conference dedicated session
  - http://www.flex-eware.org
- Extensions in active collaborative projects
  - Minalogic cluster MIND, System@tic cluster PARSEC, ITEA VERDE, ARTEMIS CHESS…
- Research agenda
  - Verification and validation (The upside of the "V" life cycle)
  - Enabling Component Assembly by « semantic typing » or «non functional" typing (e.g. resource required/ resource available » )
    - resource = MIPS, Memory, Bandwidth,…