Integrated Embedded System Development for Automotive and Aerospace Applications:

The DECOS Concepts

András Balogh, György Csertán, András Pataricza, Balázs Polgár

Budapest University of Technology and Economics

Wolfgang Herzner, Rupert Schlick, Egbert Althammer, Erwin Schoitsch

Austrian Research Centers GmbH - ARC

Martin Schlager, Bernhard Leiner

TTTech Computertechnik AG

Bernhard Huber

Vienna University of Technology

Alain Le Guennec, Thierry Le Sergent, Bruno Martin

Esterel Technologies

Neeraj Suri, Shariful Islam

Darmstadt University of Technology

Jonny Vinter

SP Technical Research Institute of Sweden

Dependable Embedded Components and Systems

(IP-Project #511764 in EU FP6 / Priority [2] IST)

- Partner (19)
  - Industry

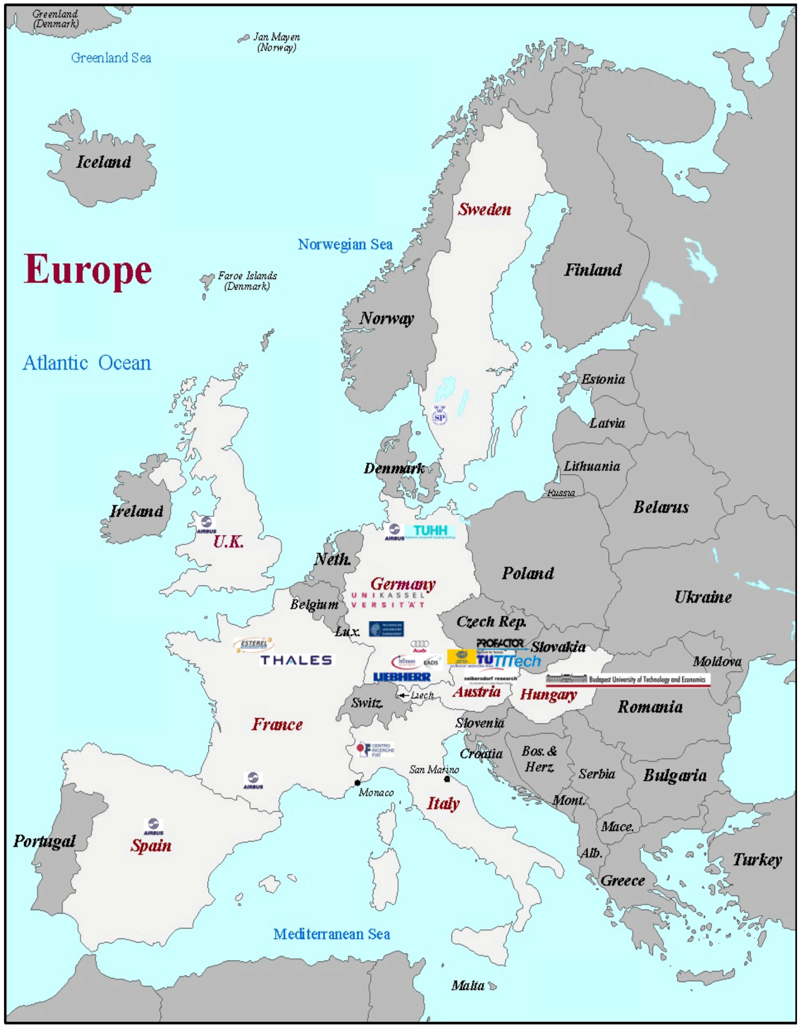
    Airbus, AEV, EADS, Infineon, TTTech, Fiat, Profactor, Hella, Liebherr, Thales, Esterel

  - Universities

    TU Vienna, TU Darmstadt,
    TU Hamburg, Uni Kassel, Uni Kiel,
    Budapest Uni of Techn. and Economics

  - Research Centres

    ARCS, SP Swedish Test. & Res. Inst.



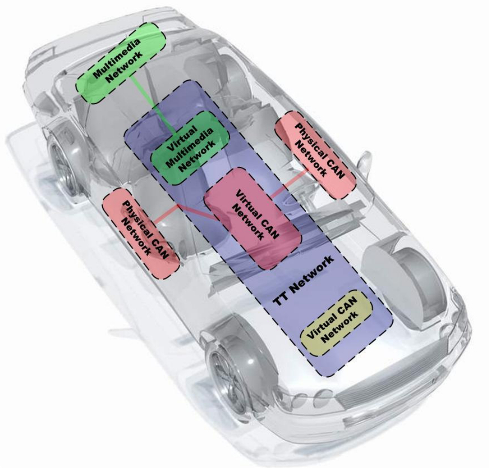
DECOS Goals

- Uniform platform for integration
  of embedded distributed (real-time) applications
  of mixed (up to highest) criticality
  - hardware reduction
  - flexibility increase

⇒ from federated to integrated systems

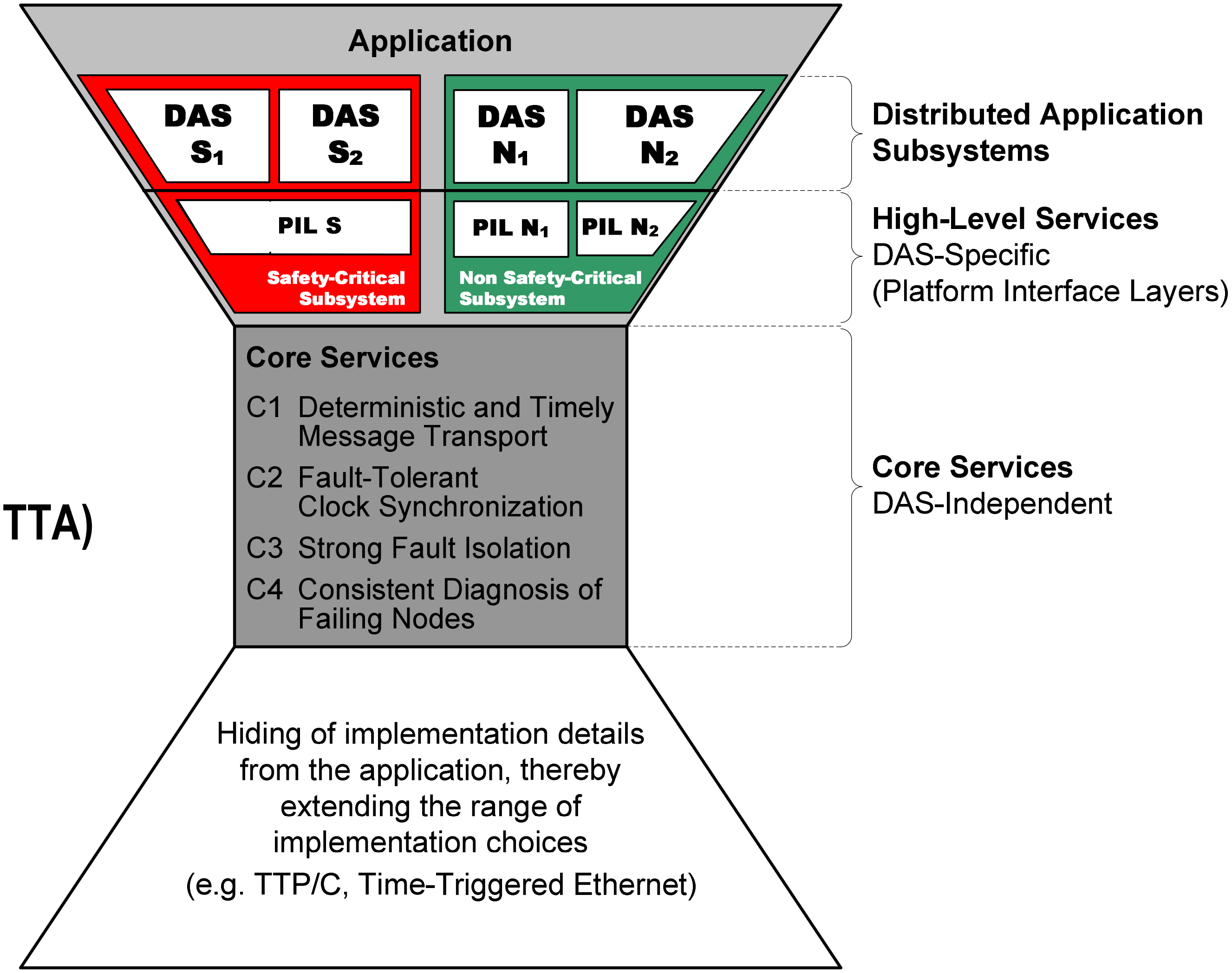
- Implication: fault-isolation of
  and non-interference between
  integrated systems has to be guaranteed

⇒ provision of appropriate
  - architectures
  - components and services
  - development and verification tools



DECOS "Wasteline" Architecture Model

- DECOS high-level services
  - Encapsulated Execution Environment
  - Virtual networks
  - Gateways
  - Diagnosis service
  - Fault Tolerance Layer
- DECOS core services
  - Prevalidated (FIT, NEXT TTA)
- Domain and Platform Independence:
  - Any core technology providing core services suffices
  - (TTP/C, FlexRay, TT-Ethernet, …)



Application

DAS S_1

DAS S_2

DAS N_1

DAS N_2

PIL S

PIL N_1

PIL N_2

Safety-Critical Subsystem

Non Safety-Critical Subsystem

Core Services

C1 Deterministic and Timely Message Transport

C2 Fault-Tolerant Clock Synchronization

C3 Strong Fault Isolation

C4 Consistent Diagnosis of Failing Nodes

Hiding of implementation details from the application, thereby extending the range of implementation choices

(e.g. TTP/C, Time-Triggered Ethernet)

Distributed Application Subsystems

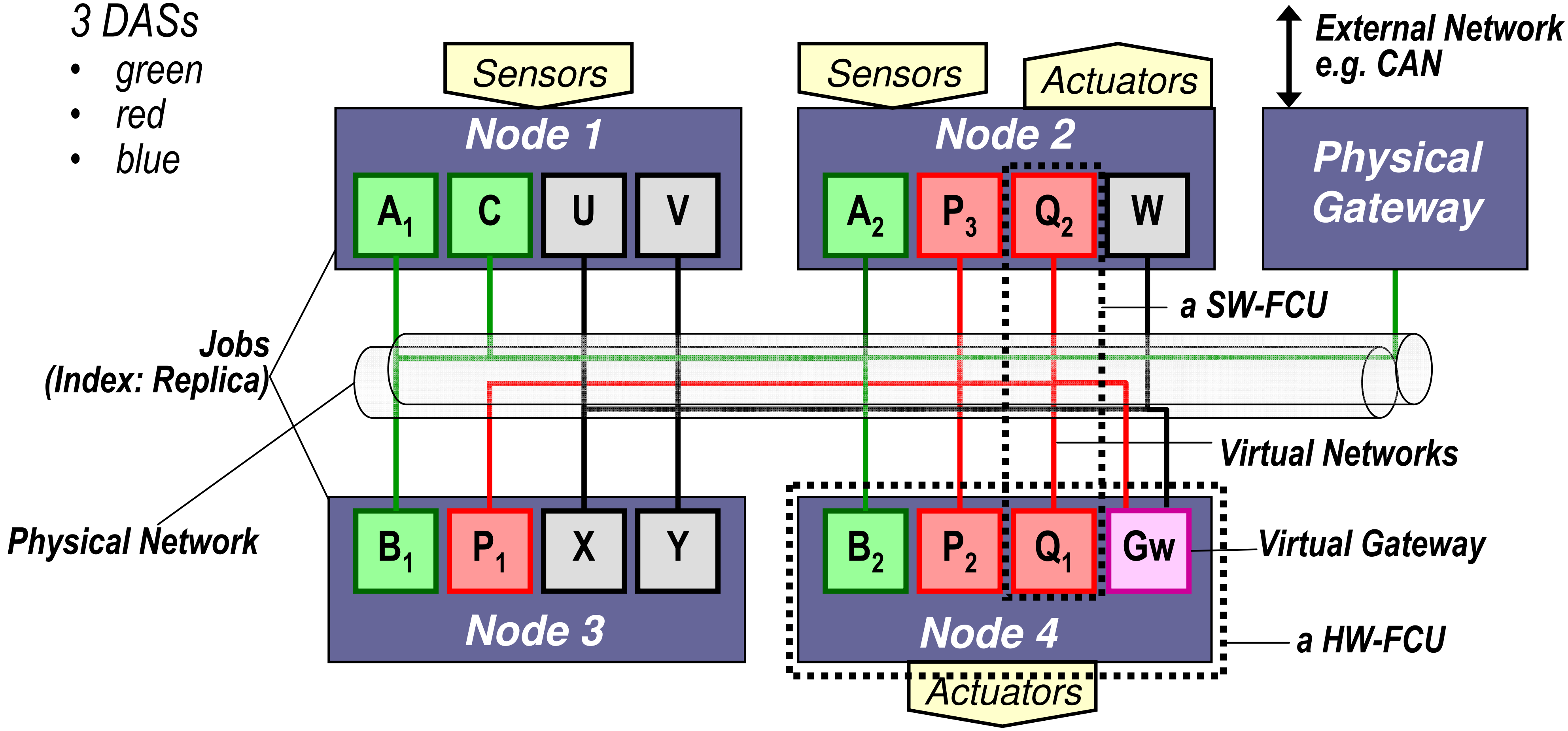
High-Level Services
DAS-Specific
(Platform Interface Layers)

Core Services
DAS-Independent

DECOS Cluster Architecture (Example)

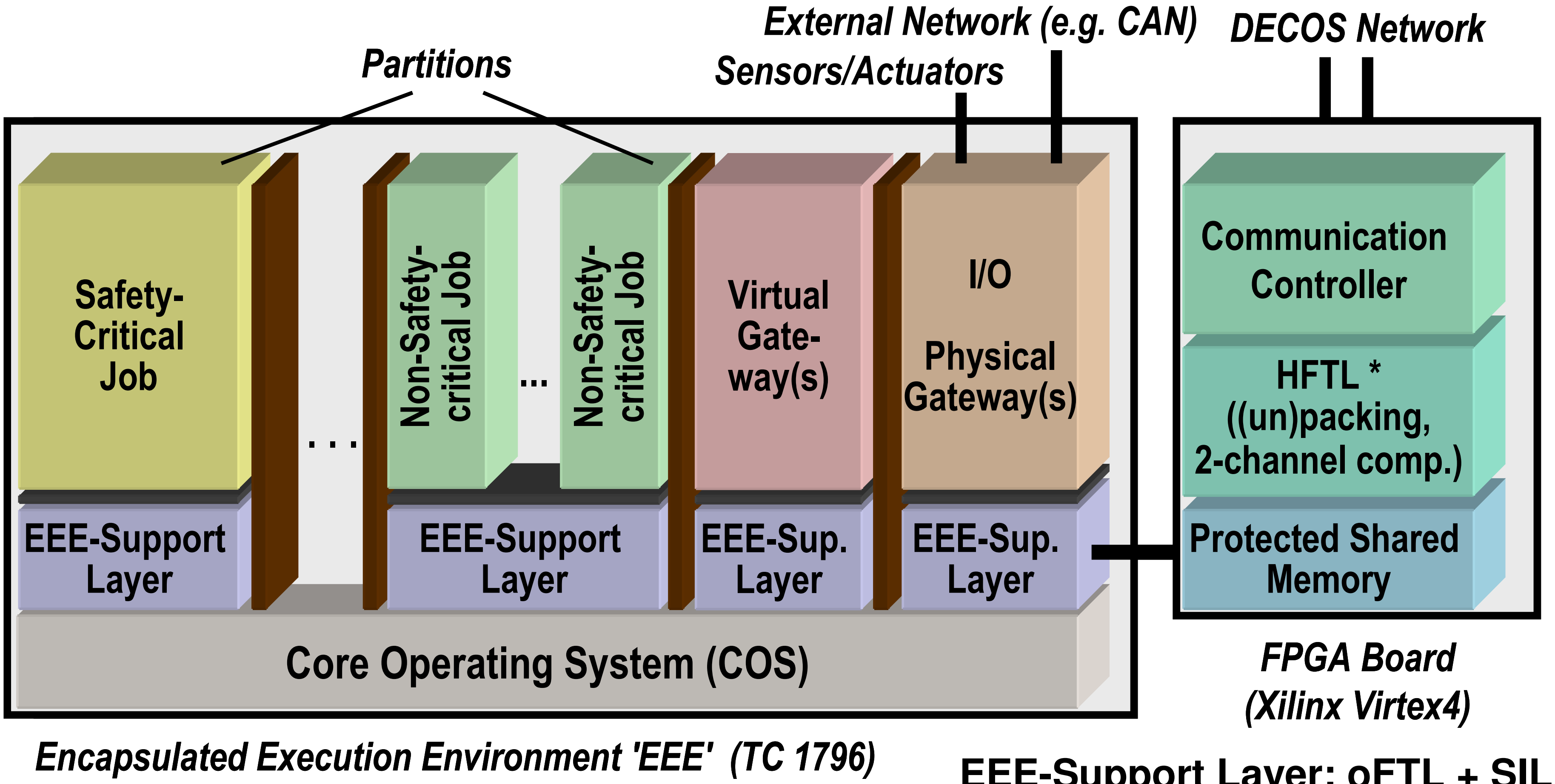
3 DASs

- *green*
- *red*
- *blue*



Fault-Containment Units (FCU): Hardware – Node, Software – Job (all replicas)

Implementation on DECOS Platform



EEE-Support Layer: oFTL + SIL
(optimized FTL + System Interface Layer)

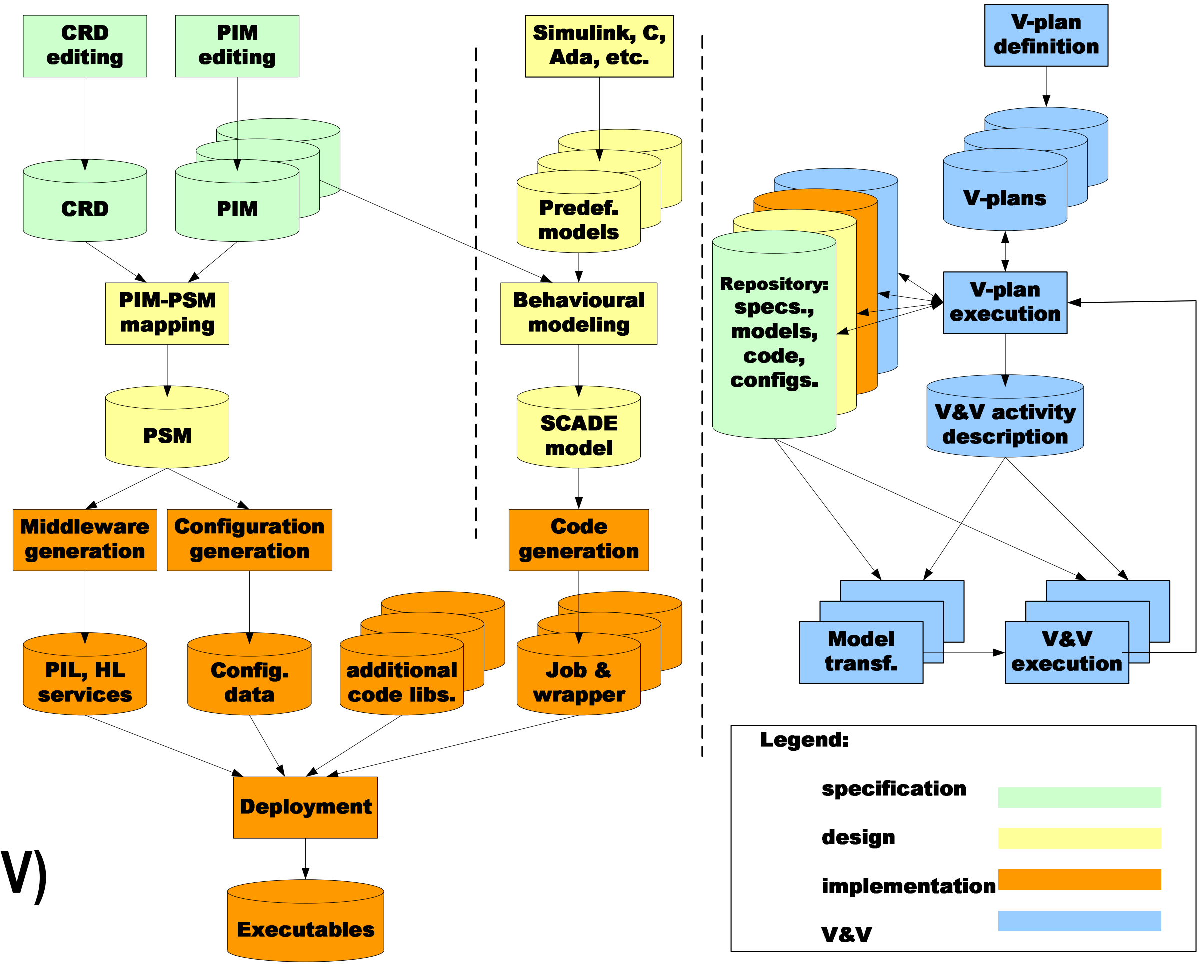
Per partition: - **memory protection**
- **execution time slot „separation in space and time“**

*** Hardware FTL**

Tool Chain: Model-Based Integrated Development Support

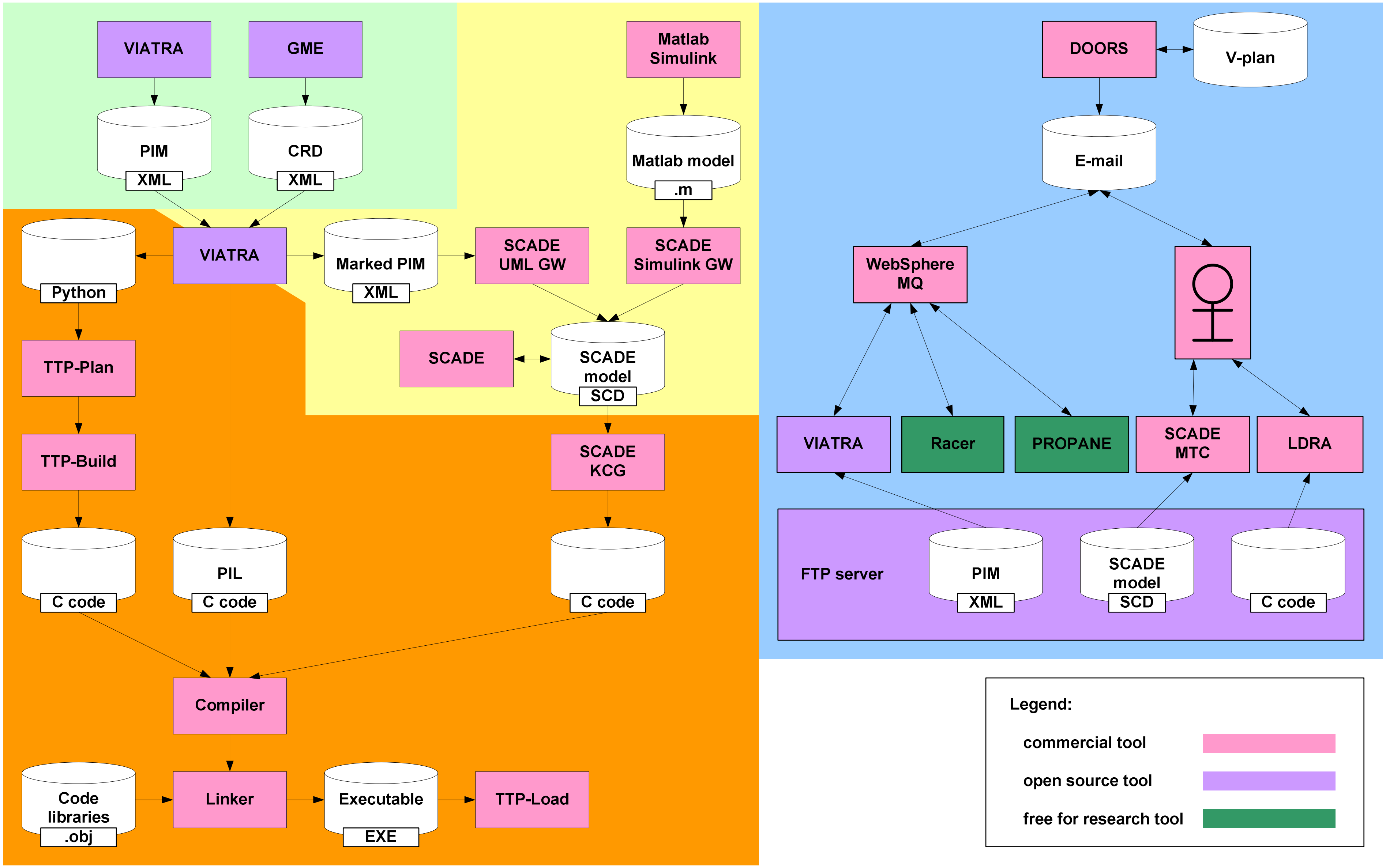
"From Requirements To Deployment"

1. Requirements
   - functional, performance, dependability
2. Cluster modelling
   - nodes, network
3. Behaviour modelling
   - of jobs
4. Configuration
   - allocation and scheduling
5. Middleware generation
   - APIs, fault-tolerance
6. Deployment
   - compile, link, download
7. Verification & Validation (V&V)
   - accompanying (Test Bench)



Tool-chain Integration

DECOS



Legend:

commercial tool 🟪

open source tool 🟪

free for research tool 🟩

Generic Test Bench – V&V Tool Integration

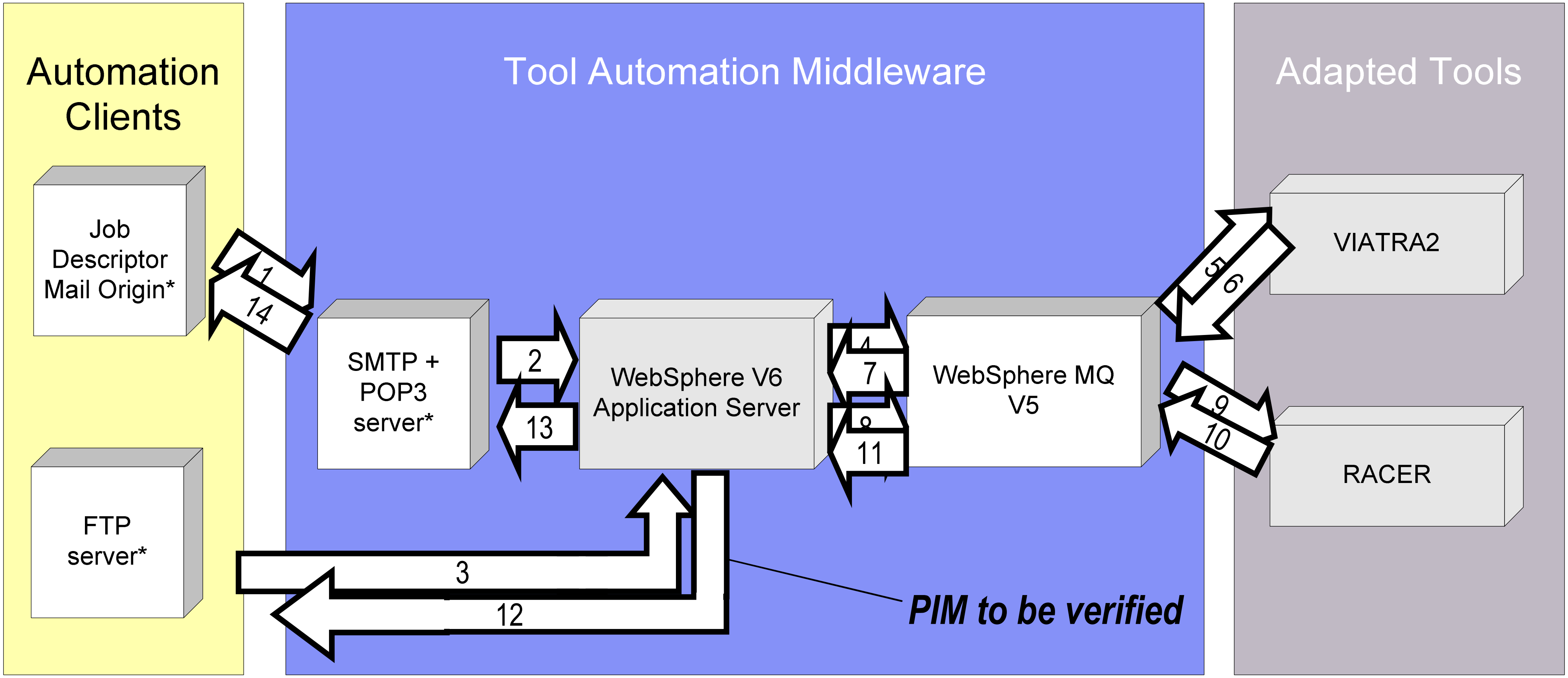
Tool integration levels

- **No external tool:** e.g. Checklist
  - Tool implemented in DOORS
- **Manually executed external tool:** e.g. PROPANE (SWIFI)
  - Start of tool in dialog ("pressing a button")
- **Automatically executed external tool:** e.g. RACER (Ontology based consistency and completeness check)
  - Start of tool by "mailing" to corresponding server (no user interaction)
- **External test bench:** e.g. EMI Hardware Test Bench
  - Tool runs on separate hardware, feedback by email/message flow

For all levels, corresponding interaction workflows provided

Example for automatically executed external tool

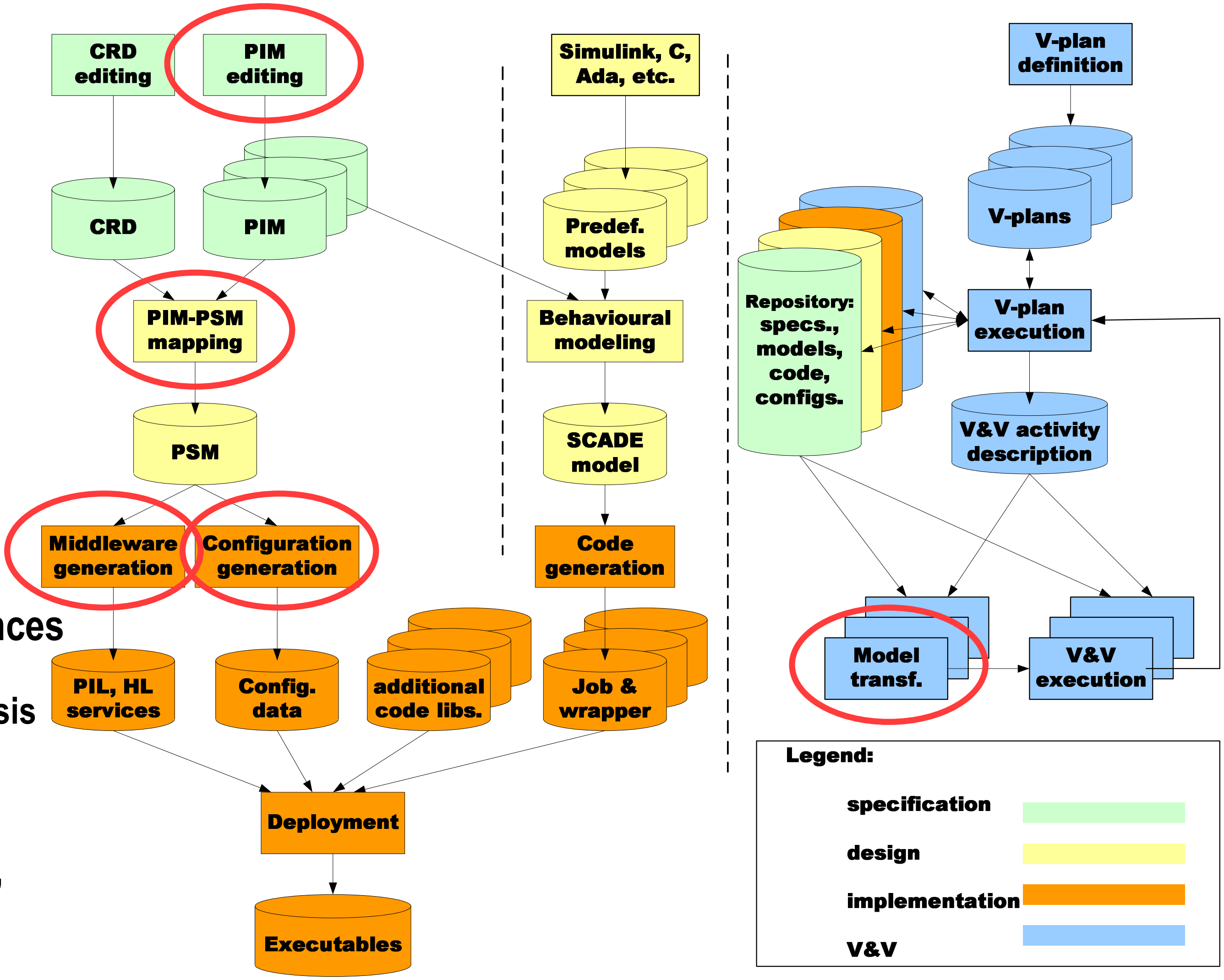
PIM-validation with Racer



VIATRA2 by BUTE

DECOS

- Modelspace
  - Multi level metamodeling
  - Base concepts:
    - entity, relation
    - inheritance, instantiation
  - Multiple domains
  - Multiple source
    - Import, export
    - ➢ Tool integration !
  - Multiple views (e.g. DSE)
- Transformation language
  - Graph transformation part
    - with patterns & rules
  - Abstract State Machine part
    - with control structures
  - Interpreted execution
  - Big abstraction level differences are easy to handle with it
    - e.g. xforms to formal analysis domains
- Implemented as Eclipse plug-in
- Open source version is available, commercial is coming soon (Spin-off SME: OptXWare)



Summary

- *Architecture* and *methodology* has been elaborated for

  specify, design, implement, validate & verify

  real-time embedded systems

  with safety-critical and non safety-critical components

  in an integrated way.

  - Model Driven Development
  - Model Driven Architecture
  - Demonstrated in automotive, aerospace, industrial control domains

- *Tool integration* is realized by

  1. well defined architecture & development process
  2. well defined extension points for development steps (Generic Test Bench for verification & validation)