Towards a GENeric Embedded System Architecture

GENESYS

H. Kopetz

June 2008

Why a *Generic Cross-Domain* Architecture?

In the domain of embedded systems, a *cross-domain* approach is needed for many reasons:

- Many issues are the same in different domains (aerospace, automotive, industrial, medical, multi-media, mobile devices):
  - Reuse of Components--Composability
  - Robustness--reduce the fragility of systems
  - Dependability (Security)
  - Energy efficiency (portable devices, heat dissipation)
  - Temporal Predictability
- Interconnection of Systems from different domains
  - Access your car with a cell phone
  - Multimedia in the car and in control systems
- Economies of Scale of the Semiconductor Industry
- Unified Development Methodology and tools to make better use of the limited human resources

GENESYS: GENeric Embedded SYStem

EU funded project to develop a generic-cross domain architecture for embedded system that meets the ARTEMIS requirements:

- Project Partners (23): TU Vienna (Coordinator), Nokia, Infineon, Thales, STMicroelectronics, NXP, Fiat, Volvo, TTTech, Ikerlan, IMEC, et al.
- Project duration: Jan 08 - June 2009

Architectural style has been completed

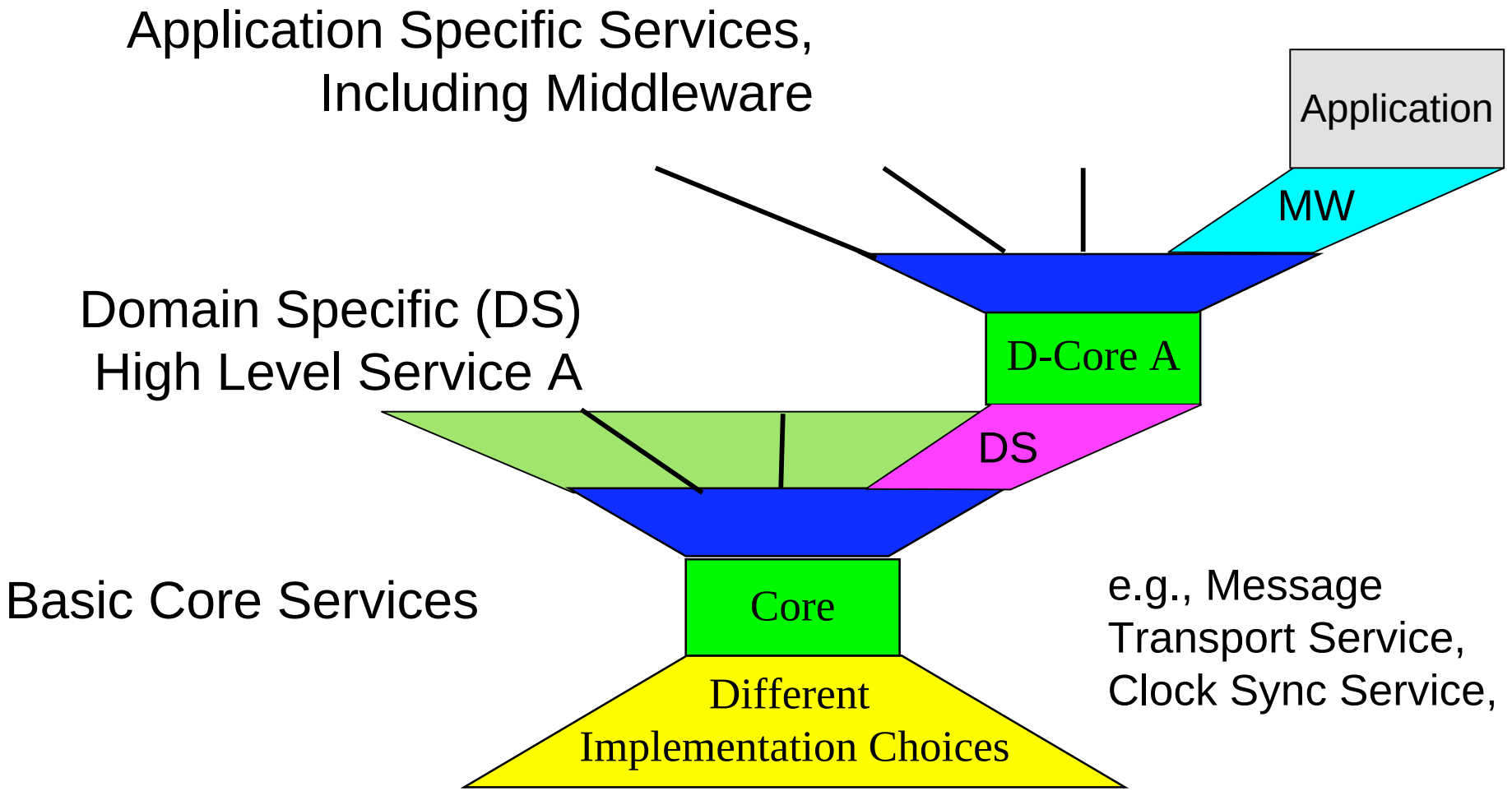
See:

http://www.genesys-platform.eu/

The Seven Key Challenges of *ARTEMIS*

- ***Composability***
- ***Networking and Security***
- ***Robustness***
- ***Diagnosis and Maintenance***
- ***Integrated Resource Management***
- ***Evolvability***
- ***Self Organization***

Genesys is a Flexible Platform Architecture



On the *Structure of Systems*

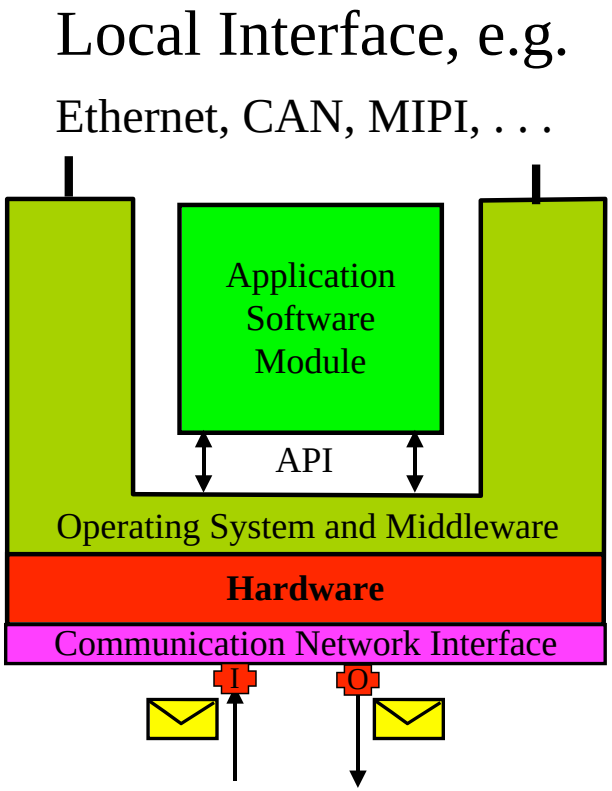
If you look at automata which have been built by men or which exist in nature, you very frequently notice that ***their structure is controlled to a much larger extent by the manner in which they might fail and by the (more or less effective) precautionary measures which have been taken against their failure.***

John von Neumann, Theory of Self-Reproducing Automata, Urbana, University of Illinois Press, 1956

Architectural Principles of GENESYS

- *Strict component orientation*: effective fault containment
- Separation of Computation from Communication: *Messages*
- *Hierarchical Structure*: Three Integration Levels (complexity)
- *Common time base* of appropriate precision at all nodes (IP-cores)
- *Deterministic* Core Services to support traceability and TMR
- Integrated *Diagnostic*, *Robustness* and *Security* Services
- Support for *Model-based Design*
- *Fate sharing* to achieve compatibility with the INTERNET
-
- (about 30 architectural principles)

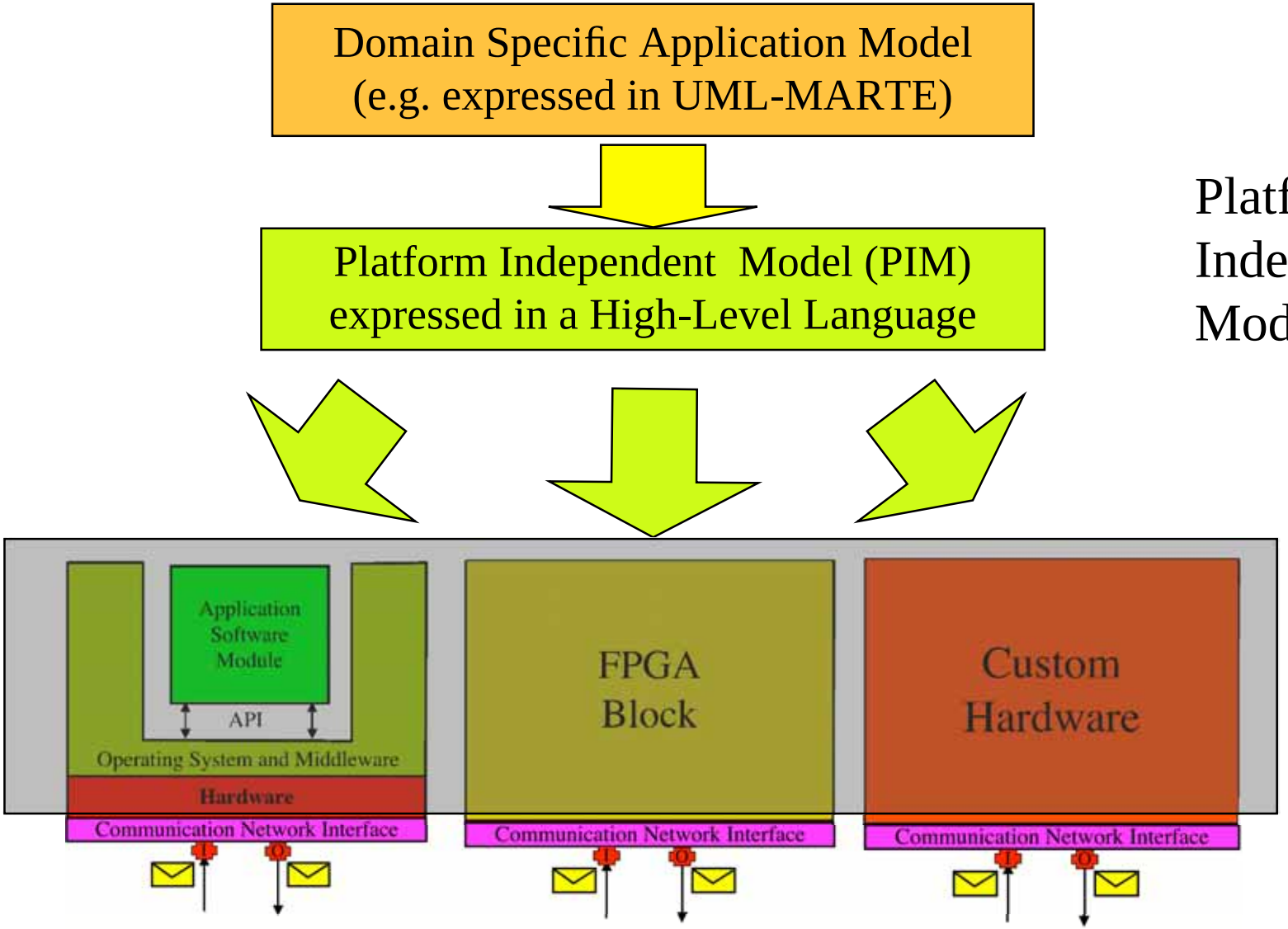
The Notion of a Component as a HW/SW-Unit



A Genesys component

- is a hardware/software unit of defined application functionality (an *IP Core*)
- contains a linking interface (LIF) which is fully specified in the domains of value and time
- is a fault-containment unit with specified error-containment boundaries
- has a defined *restart state* at its restart instant (*cyclic*)
- Is aware of the common time
- can have (unspecified) local interfaces

Model Driven Design: From the PIM to the PSM



Platform Independent Model

Platform Specific Model

Unidirectional Multicast Message Passing

Three types of messages:

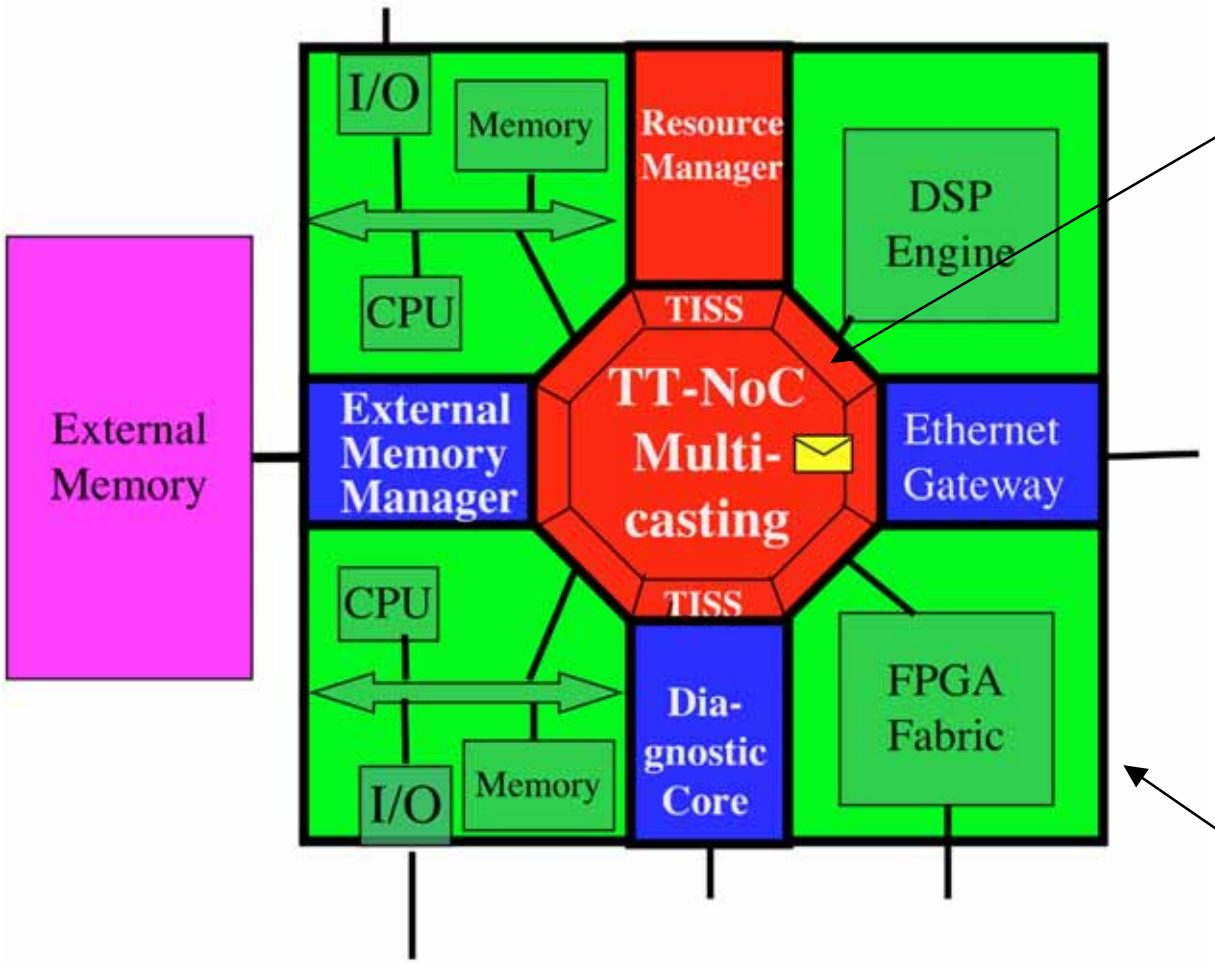
- **Event-triggered Messages** with *exactly once semantics* and queues at the sender and at the receiver (core)
  - Guaranteed bandwidth
  - Best-effort bandwidth
- **Time-triggered Messages** with *up-date in place* and *non-consuming read (*core*)*
- **Data-streams** with support for on-the fly processing, including water marking (optional core service)

Three Integration Levels for Complexity Management

- **Chip Level:** *IP Cores* are integrated by sending messages across a deterministic Network-on-Chip
- **Device Level**: *Chips* are integrated to form a device.
- **System Level**: *Devices* are integrated to form a system
  - Closed system
  - Open system (devices come and go dynamically)

Gateway components are used to link different levels. A gateway has two different LIFs, one *internal* to the lower level and one *external* to the higher level

Genesys System-on-a-Chip



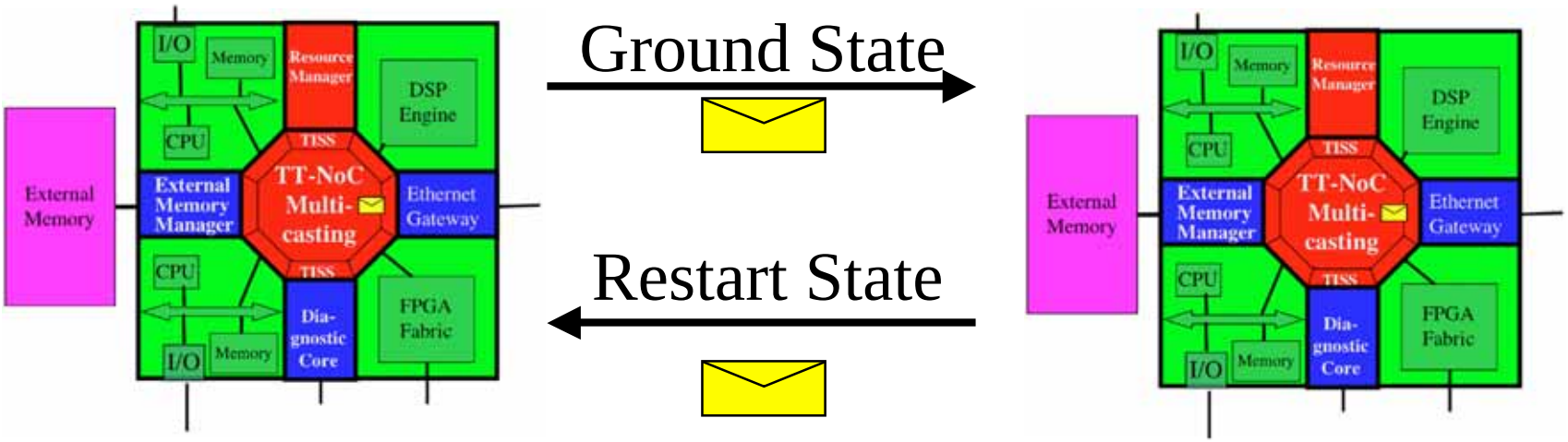
TISS: Trusted Interface Subsystem provides the LIF at the Chip Level.

Local Interface

(unspecified at the Chip level)

IP Core

Robustness



Component sends its ground state regularly to the ground-state monitor

Ground state monitor performs error detection and restart with a state-estimated restart state

Conclusion

- The Genesys project aims to specify the frame-work for a cross domain European Architecture for Embedded Systems--from safety-critical systems to mobile devices operating in an open environment.
- The basic building blocks of Genesys are components (hardware/software units) that exchange information via messages only.
- The concern for dependability is a driving force for Genesys.