The ARTEMIS Technology Platform

Hermann Kopetz, TU Wien, Austria

ADVANCED RESEARCH & TECHNOLOGY FOR EMBEDDED INTELLIGENCE AND SYSTEMS

ARTEMIS

- *ARTEMIS* is a public private partnership (PPP) for the promotion of research in the field of embedded systems in order to strengthen European leadership in this field--possible in the form of a JTI (joint technology initiative).
- *ARTEMISIA* is the legal entity which has been established to manage ARTEMIS projects.
- The funding level for ARTEMIS projects is envisioned to be between 500 Mio € -- 1000 Mio € per year, starting in 2008. It is expected to come from EU, national and industry sources.

Founding Members

- Thales
- ST Microelectronics
- Philips
- Nokia
- Daimler Chrysler

Outline

- Introduction
- Methodology
- The Seven Research Priorities
- Conclusion

URL: http://www.artemis-office.org/

ARTEMIS

- On June 30, 2005, ARTEMIS has published a *Strategic Research Agenda (SRA)* that outlines the objectives and the research topics that need to be investigated in the field of embedded systems.
- One important research domain of the SRA of ARTEMIS relates to the development of generic ***reference designs and architectures*** for embedded systems (ARTEMIS expert group on *Reference Design and Architectures*).

SRA-- Reference Design and Architectures

According to the SRA of ARTEMIS (p.16) the objective of the Expert group on ***reference designs and architectures*** is to

Identify research needs that lead towards the creation of a generic platform and a suite of abstracts components with which new developments in different application domains can be engineered with minimal efforts.

Members of the Expert Group

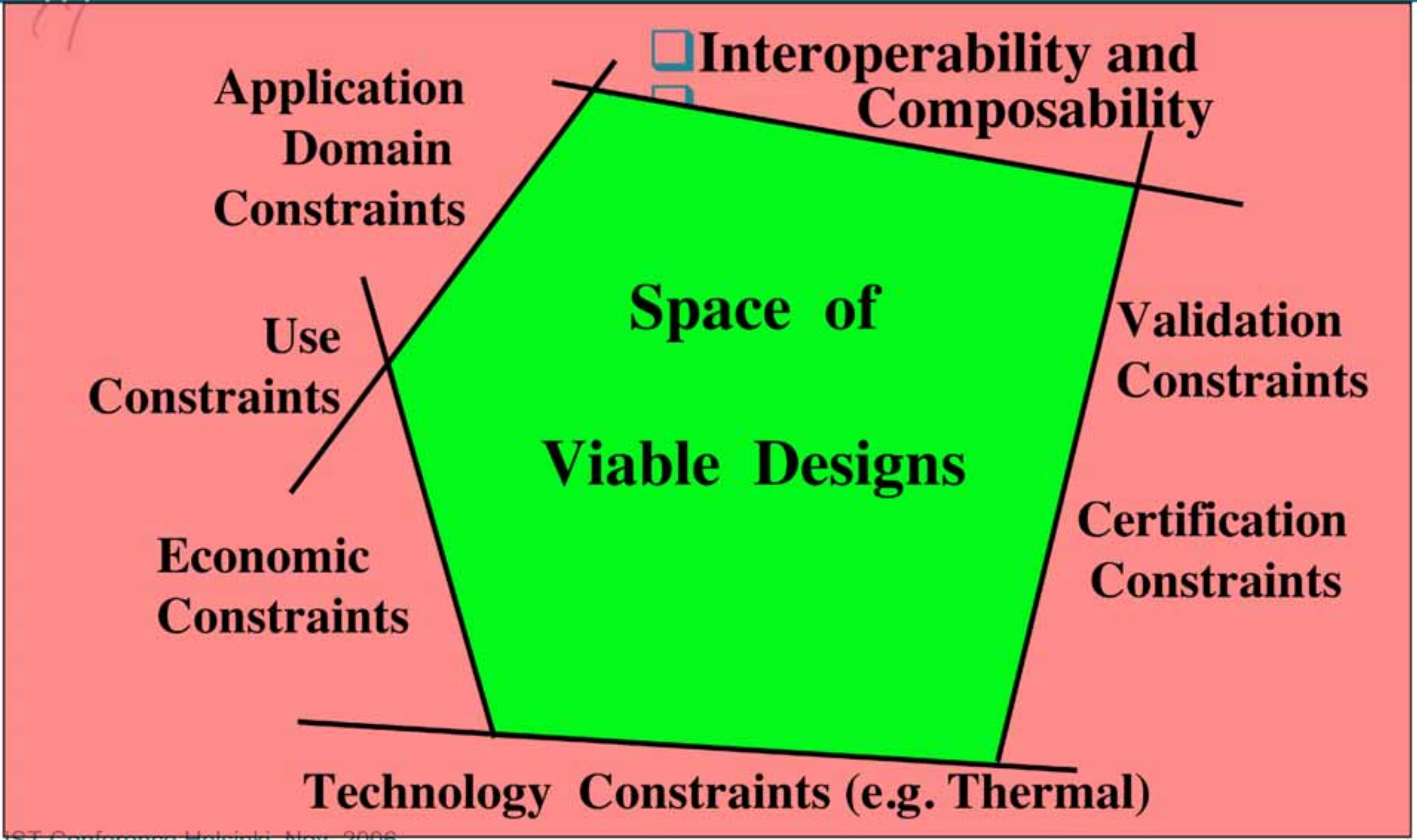
- Sergio **Bandelli**, European Software Institute
- Andrei **Bartic**, IMEC
- Christian **Bettstetter**, Alpen-Adria-Universität Klagenfurt
- Michael **Borth**, DaimlerChrysler AG
- Ed **Brinksma**, Embedded Systems Institute
- Tom **Clausen**, CEC
- Jean-Luc **Dormoy**, Commissariat à l'Energie Atomique
- Gilbert **Edelin**, THALES
- Christian **EL Salloum**, TU Vienna (Editor)
- Alun **Foster**, STMicroelectronics
- Laila **Gide**, Thales
- Magnus **Granström**, Volvo
- Riccardo **Groppo**, CRF
- Peter **Heidl**, Robert Bosch GmbH
- Knut **Hufeld**, Infineon
- Hermann **Kopetz**, TU Vienna (Editor)
- Kimmo **Kuusilinna**, Nokia
- Vera **Lauer**, DaimlerChrysler AG
- Per **Lindgren**, Lulea University of Technology
- Roman **Obermaisser**, TU Vienna
- Ton **Peerdeman**, Thales
- Ian **Phillips**, ARM
- Peter **Puschner**, TU Vienna
- Christophe **RENAUD-BEZOT**, Thales
- Herbert **Rödig**, Infineon
- Fulvio **Rusina**, COMAU
- Hans **Schurer**, Thales
- András **Tóth**, Ericsson AB
- Theo **Ungerer**, University of Augsburg
- Mateo **Valero**, Technical University of Catalonia
- Sjir **van Loo**, Philips Research
- Johan **Vouncks**, IMEC

Methodology to Arrive at the Research Priorities

The working group has adopted the following procedure to arrive at the research priorities:

1. Collection of concrete *requirements* and *constraints* that were considered relevant from the point of view of different application domains
2. Classification of the collected requirements and constraints from the point of view of different application domains.
3. Evaluating the state of the art in meeting these requirements and constraints
4. Identification of research needs and priorities.

Good Design Comes from the *Proper* Constraints



Example of a Constraint

1.16 Composability Constraint: Stability of Prior Services

Description: The architecture has to assure that a validated service of a subsystem, both in the value domain and time domain, must not be refuted by integrating the subsystem into a larger system.

Rationale: The *stability-of prior service constraint* is essential to allow independent design, development, and verification of a subsystem.

Research Challenge:Definition of a framework that eliminates all unintended side effects of integration without an excessive performance penalty.

- More than 150 requirements and constraints have been captured
- Each of these has been classified by a given application domain according to
  - Essential
  - Very desirable
  - Desirable
  - Don’t care
  - Forbidden
- More than twenty classifications have been received (they are all contained in Annex II of the extended SRA report).

The Technology Landscape--Hardware

- The limits of *Moore’s* law are becoming visible: power dissipation, physical feature size, reliability.
- The performance increase of a single processor from one generation to the next is proportional only to the square root of the increase in silicon area (*Pollack’s* rule).
- Multi-computer chips (SoC) are appearing. The development cost of such a chip can pass the 100 Mio $ wall--mass markets are needed to justify this level of investment.
- The transient and permanent failure rates of sub-micron devices is increasing [both single-event upset (SEU-memory) and single-event transient (SET-logic)].

Mitigation of Soft Errors by Architectural Means

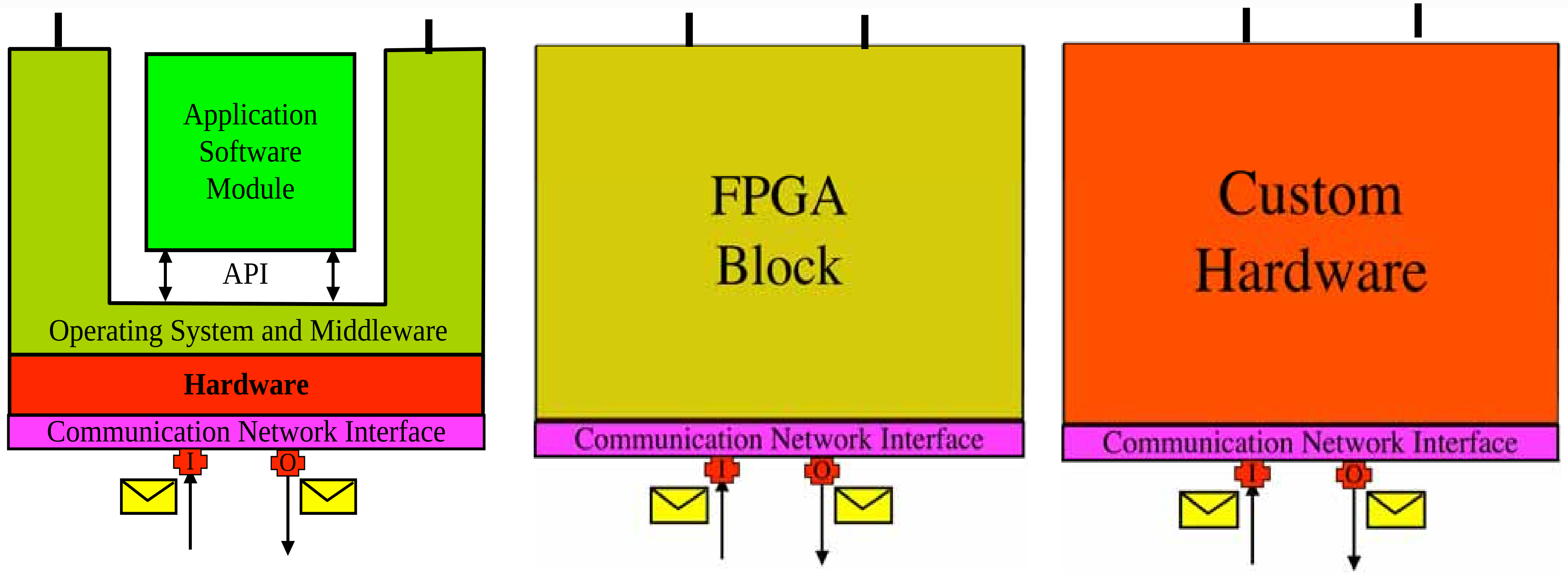
Architectural means to mitigate the consequences of component failures might become a necessity when using the upcoming submicron devices, as stipulated in the latest *2005 International Roadmap of Semiconductors p.6: "Relaxing the requirement of 100% correctness for devices and interconnects may dramatically reduce the costs of manufacturing, verification and test. Such a paradigm shift is likely* ***forced in any case by technology scaling****, which leads to more transient and permanent failures of signals, logic values, devices and interconnects."*

The Seven Key Challenges of *ARTEMIS*

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

Different Types of Components

Local Interfaces--Open Components



Application Software Module
API
Operating System and Middleware
Hardware
Communication Network Interface
I
O
FPGA Block
Communication Network Interface
I
O
Custom Hardware
Communication Network Interface
I
O

The Communication Network Interfaces of all three different types of system components should have the same *syntax*, *timing* and *semantics*. For a user, it should not be discernible which type of system component is behind the Interface.

Composability

- Techniques for the precise specification of the interface behavior of a component (value and temporal) both at the syntactic and semantic level, including formal and semiformal interface models that are expressed in tool-processable metadata with precise semantics.
- Encapsulation of self-contained subsystems, comprised of assemblies of communicating components, such that any unintended interference (temporal, value) between unrelated subsystems is excluded by architectural means.
- Conceptualization and characterization of the aggregate properties of communicating component assemblies in order to form a *new self-contained subsystem*, thus elevating the level of abstraction to a higher level.

Networking and Security

- Architectural support for *structured name spaces*. Dynamic and trusted *name address binding* such that moving entities can maintain a disruption-free connection in a mobile environment. Routing algorithms that are based on *names* rather than *addresses*.
- Communication protocols that give the user the option to make a choice between latency, jitter, throughput, determinism, reliability, security, and power efficiency.
- Architectures that provide a *global time service* and predictable transport mechanisms for *pulsed real-time data streams* within control loops across different types of networks.

Networking and Security (ii)

- Ad hoc wireless networks that provide sustained and trusted performance to smart sensors, actuators and other embedded system nodes in harsh environments (e.g., *car-to-car* and *car-to-infrastructure* networks).
- Automatic security management (authentication, authorization, privacy, confidentiality of information, protection of intellectual property, digital rights management) of trusted embedded environments considering the limited resources (energy, bandwidth, power) of embedded nodes.
- Secure authentication infrastructure for embedded entities that cannot be forged by malicious intruders and can be used by the authorized applications to control access privileges.

Robustness

- Development of fault-tolerant architectures that mitigate the effects of *soft errors* (e.g., the transient corruption of a single bit by low-energy cosmic neutrons) at the system level. Architectural support for error masking, such as error-correcting codes or triple-modular redundancy (TMR), for critical system services.
- Self-configuration, reconfiguration and self-commissioning of systems in a novel environment without explicit user interaction.
- State-aware design techniques that always keep track of the *critical system state* and provide mechanisms to detect and repair corrupted state within a short error-detection latency.
- Quantitative fault models, including failure modes and failure rates for sub-micron VLSI devices, components and subsystems.

Diagnosis and Maintenance

- Provision of a framework that supports the observation of component behavior and the identification and isolation of faulty components. Independent on-line checking of *pre-conditions of input data* and *post-conditions of output data* of components.
- Architectural techniques for in-system test generation and on-line testing without the probe effect. Provision of the capability for built-in test (BIST) at the system level.
- Architectural means and novel algorithms for the on-line analysis of diagnostic data to reduce the diagnostic data volume and arrive at reliable information about the current *health state* of the components.
- Support for non-intrusive on-line auditing: error detection, intrusion detection, detection of misconfigurations. Provision of an auditing framework that is customizable by the user.

Integrated Resource Management

- Development of a framework where multi-voltage zones and multi-clocking zones can coexist on a single system-on-a-chip (SoC) and where the voltages and clock speeds in the individual zones can be controlled by software.
- Algorithms that accurately estimate system-level power requirements, both the peak power and the integral power needed to complete a task.
- System-level resource management algorithms (power, execution time, bandwidth, memory) that dynamically allocate resources to tasks such that the deadlines of all time-critical tasks are met and the given budgets for resource–usage (power) are observed.
- Power-aware algorithm design: Development of algorithms and execution environments that optimize the power usage. Power aware scheduling algorithms.

Evolvability

- *Generic platform evolvability* versus *artifact evolvability*: Finding a proper level of abstraction for the architectural style such that the architectural style is much more stable than the artifacts that are instantiated within this architectural style.
- Legacy management: Integration of an existing application into a novel architecture and the related issue of integrating a novel architecture into an existing application.
- Technology obsolescence management: In a number of scenarios (e.g., aerospace industry, process control) the embedding system (the airplane or plant) has a much longer life-cycle than the embedded system.

Self Organization

- Emergence of complex behavior out of an assembly of simple components that are context aware and act autonomously within their limited context.
- Situational awareness, such as location, time, power and discovery of cooperating and adversary entities and the capability to generate autonomously plans for goal attainment, taking the situational awareness as input.
- Decentralized management: autonomic embedded systems that are capable to install and configure themselves, learn about their context, maintain themselves and reconfigure themselves in case of failing components without any user intervention.

Two Types of Research

(i) **Architecture Research**: Exploration in and prototypical design of *generic platforms* that address these seven key challenges in the best possible manner. This research should both be of conceptual and experimental character.

(ii) **Supporting Research**: Deeper research in each of the main research areas in order to gain a better understanding of the issues involved and learn about the limitations and opportunities of a synergistic combination of desirable system properties.

A Way to Proceed

(i) **Requirements and Constraint Capture: done**

(ii) **Generic Platform Conceptualization**: In this phase first blueprints for a number of candidates for the envisioned ARTEMIS *generic platforms* are developed.

(iii) **Selection of Candidates**

(iv) **Prototype Implementation of an Architecture Demonstrator**: In this phase, prototypes of the selected generic platform candidates are implemented.

(v) **Prototype Evaluation and Consolidation:** In this phase a comparative evaluation of the prototypes implemented in the previous phase is conducted and candidates that form the basis of the ARTEMIS generic platforms are selected.

(vi) **Industrialization**

Conclusion

- We are in a period of dramatic change--such a period offers great opportunities, but some of the present views and approaches have to be reconsidered.
- There are many common requirements and constraints across different application domains of embedded systems.
- The economic scenario of the semiconductor industry requires a common approach across application domains.
- ARTEMIS tries to establish such a common approach by creating a public-private partnership which supports a common strategic research agenda (SRA) in order to maintain the European lead in the embedded system area.