
MPSOC Challenges

- The Artemis MPSOC Experience -

Tiberiu Seceleanu

ABB Corporate Research
Västerås, Sweden

The A-MPSOC Group

- Contributors:
 - Tughrul Arslan, University of Edinburgh
 - Luca Benini, University of Bologna
 - Koen Bertels, Delft University of Technology
 - Phillippe Bonnot, Thales
 - Salvatore Carta, University of Cagliari
 - Thierry Collette, CEA LIST
 - Gilbert Edelin, Thales
 - Laila Gide, Thales
 - Kees Goossens, NXP
 - Axel Jantsch, Royal Institute of Technology
 - Ahmed Jerraya, CEA
 - Rudi Lauwereins, IMEC
 - Jan Madsen, Technical University of Denmark
 - Bob Malcolm, ideo Ltd.
 - Sakir Sezer, Queen's University Belfast
- Rapporteurs:
 - Tiberiu Seceleanu,
 - Hannu Tenhunen, University of Turku

MPSOC

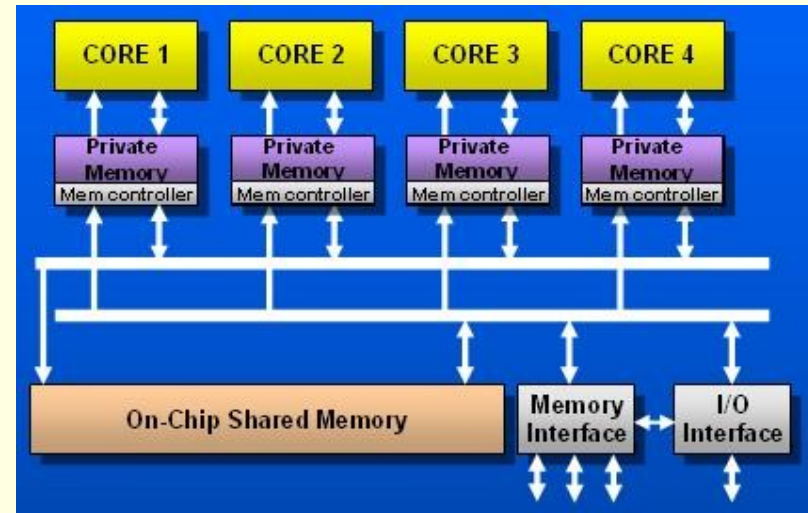
- Single chip architecture that contains

- Processing elements:

- Multiple,
- Heterogeneous,
- Flexible

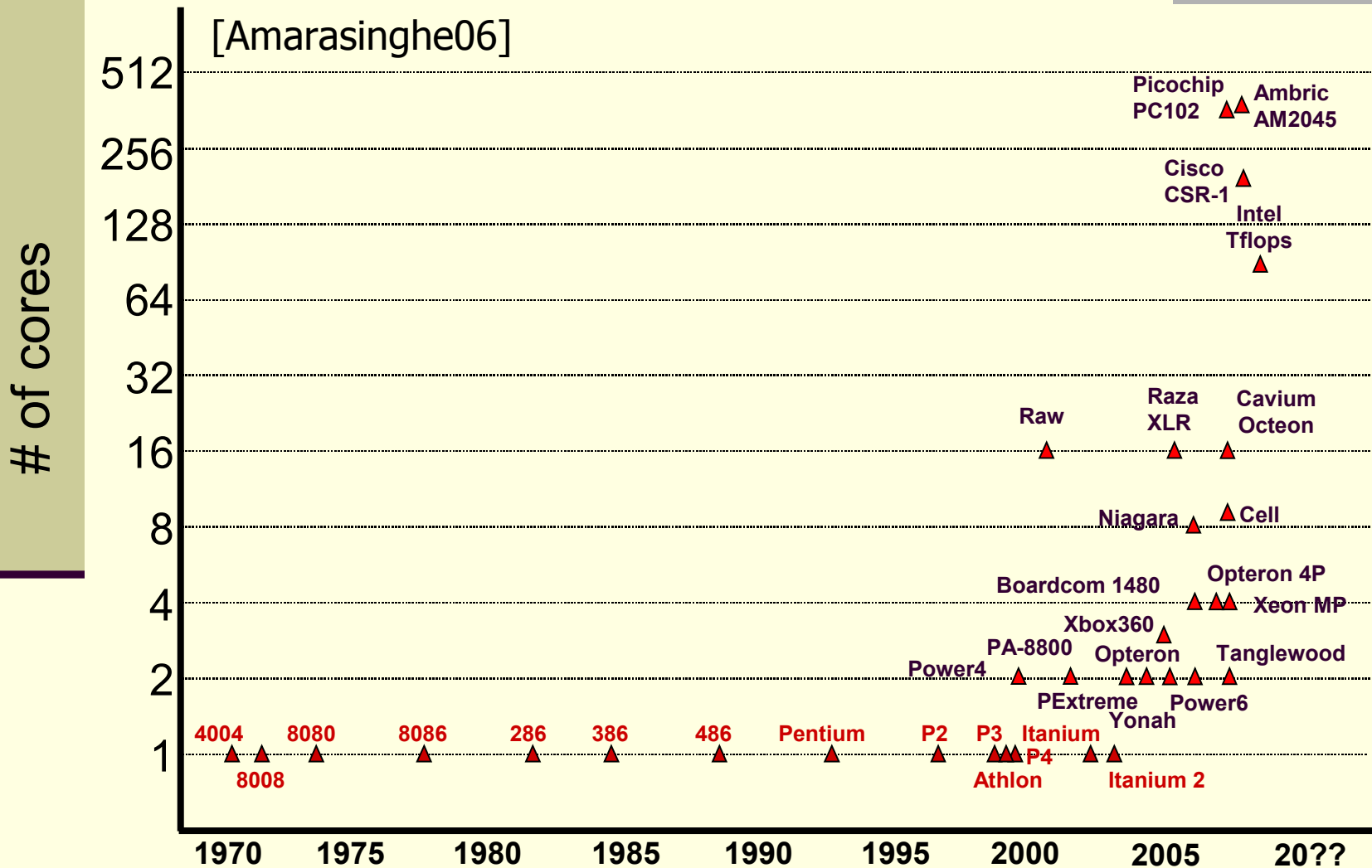
- A memory hierarchy

- I/O components.

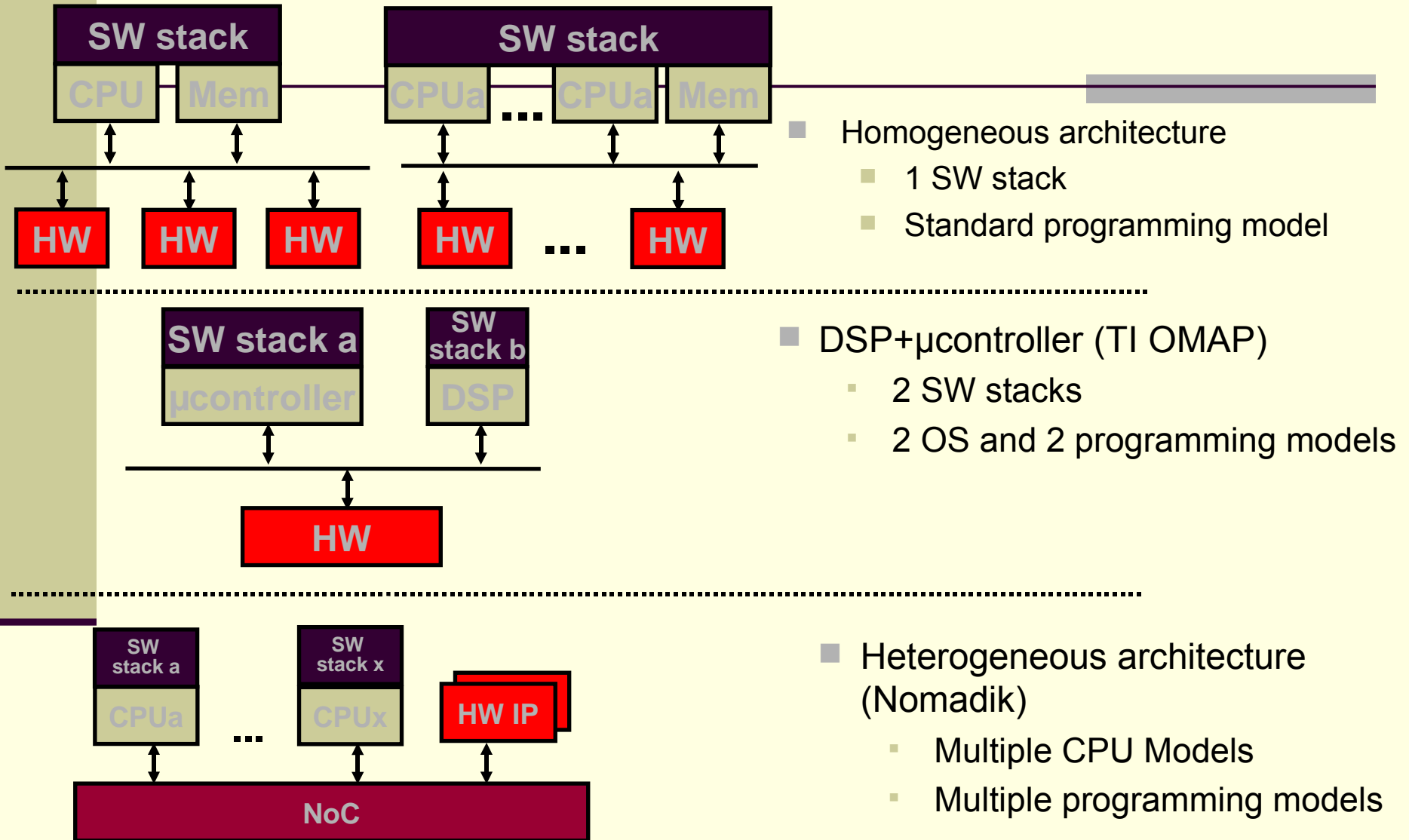


- Components are linked to each other by a flexible on-chip interconnect structure.
- Focus on design issues and utilization scenarios

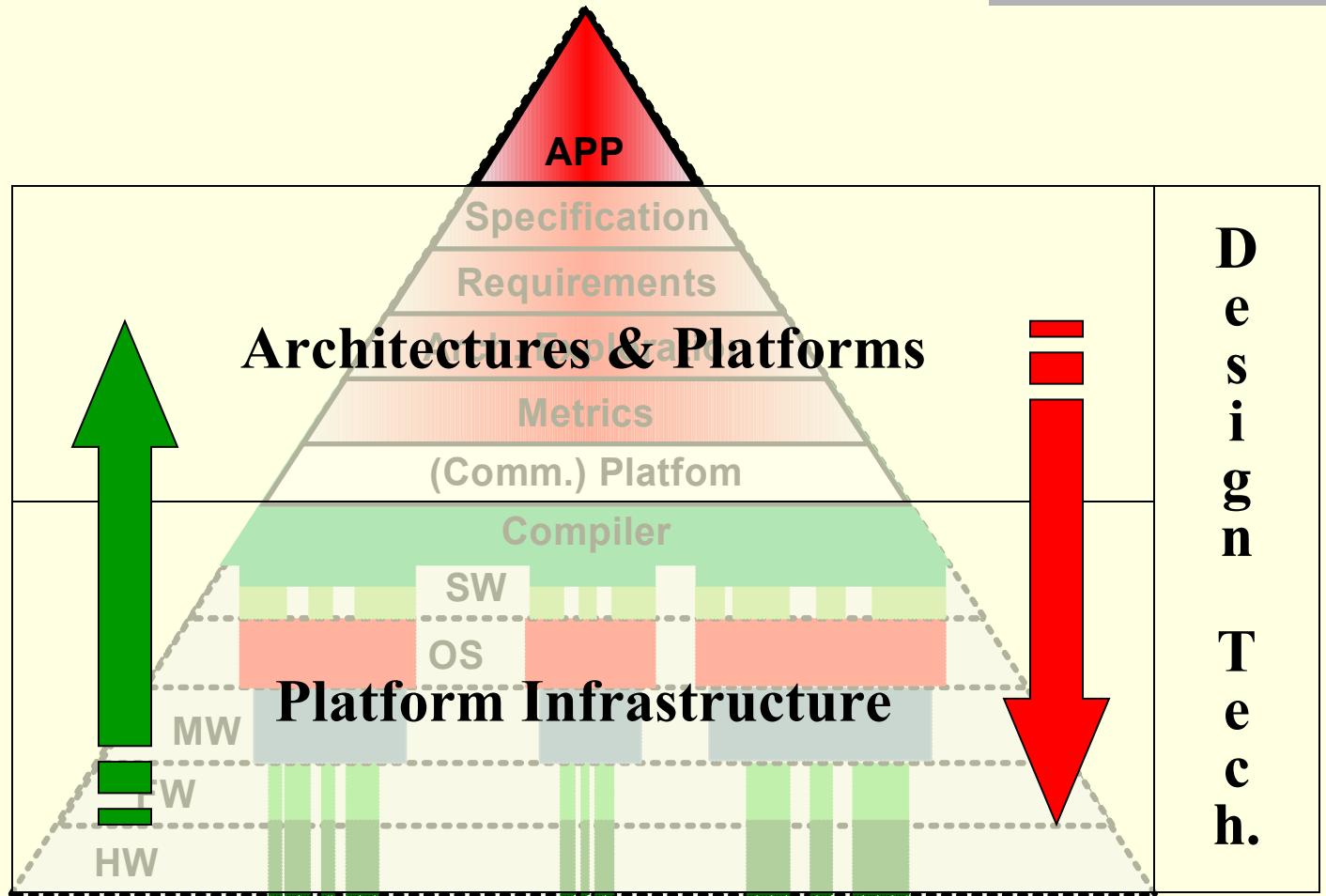
Multicores Are Here



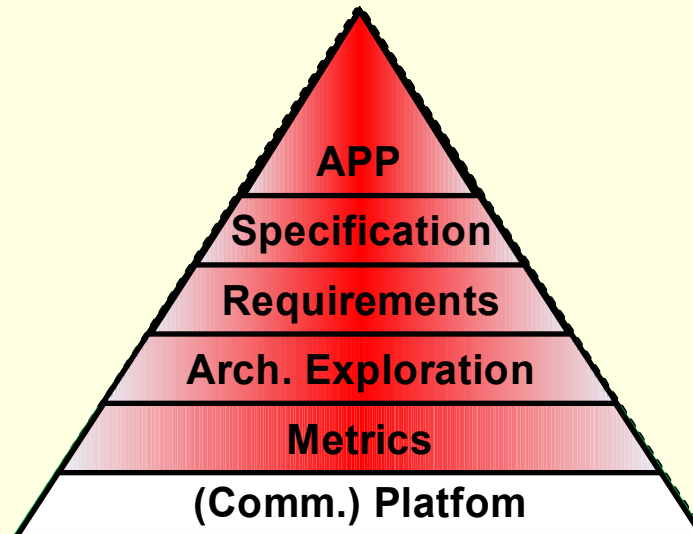
MPSoC Architecture Trends



Approach



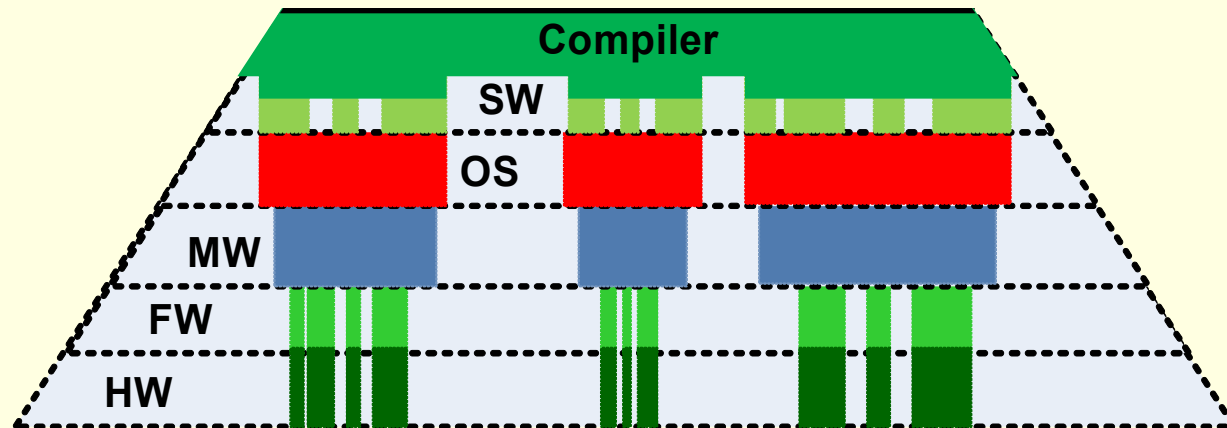
Architectures and Platforms



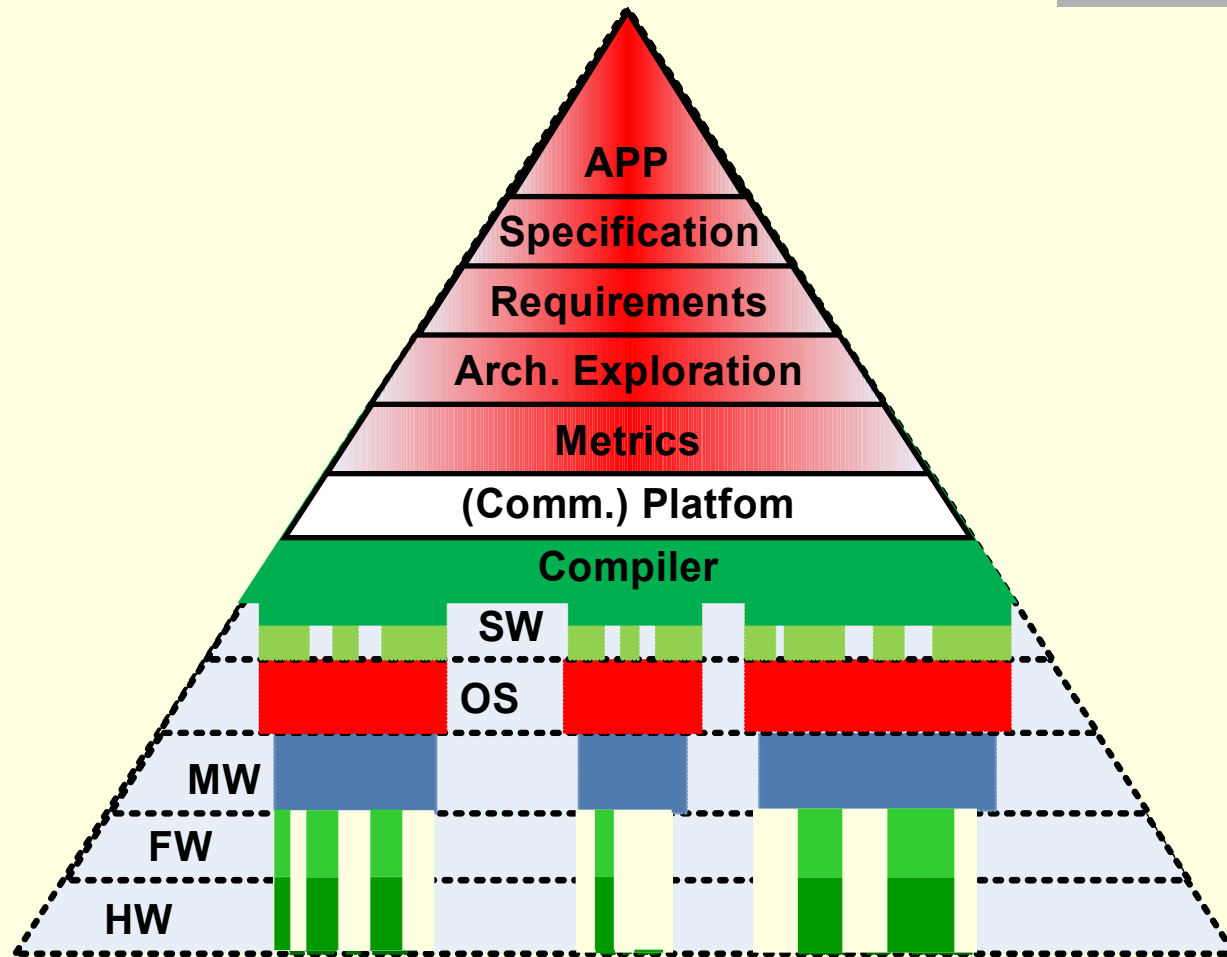
- Platform abstraction & characterization
- Concurrency modeling.
- Reconfiguration: Self-X, “programmed”-X

Platform Infrastructure

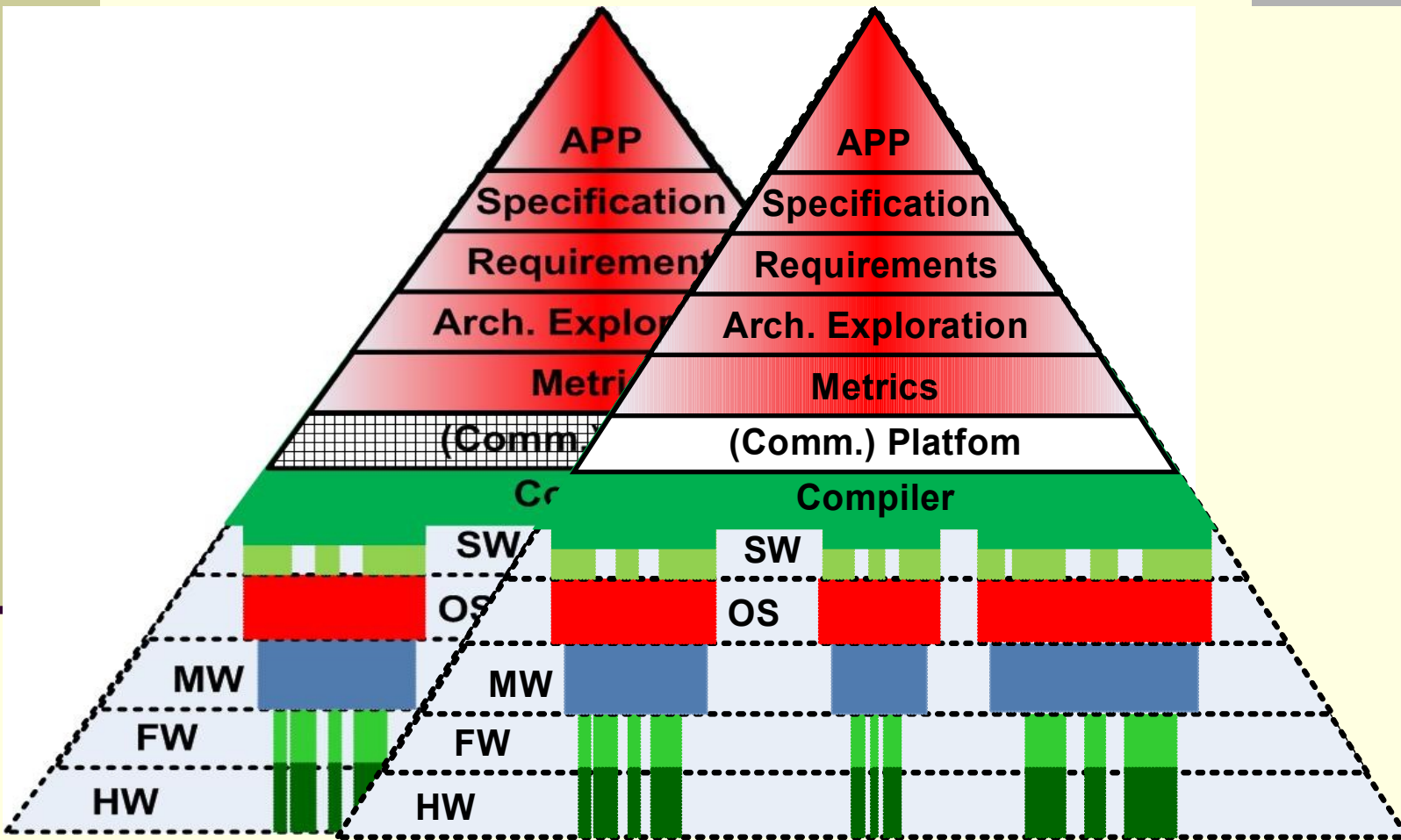
- communication backbone (HW)
- SW-HW (FW)
- SW-SW (MW, OS, SW)



Design Stages



Design Stages



Organization

■ MPSOC *Priorities*

- Quality features desirable in every MPSOC
 - At platform level
 - At component level
 - At infrastructure level
- Required capacities, which, even researched in “isolation”, would lead to “global” advances

■ MPSOC *Challenges*

- Open R&D topics
- Issues to be addressed by each priority
- Providers of R&D topics

MPSOC Priorities - Compositionality

- The composability of *functionality, communication* and *memory usage* is essential to ensure limited and controlled interference between applications.
- Composability of functions, communication and memory usage is the pre-requisite for *efficient functional verification, efficient performance analysis* and *predictable run time behavior*.
- Fully composable applications + predictable behavior → safe to adding new features and applications under end-user control.
- *Hard composability* allow to mix security and safety critical applications with less critical applications on the same platform.
- System design will be largely done through composition using standard, off the shelf components
- Runtime scalability (or any other kind of adaptation) will require standard interfaces and mechanisms for extending and adapting the system structure to runtime conditions

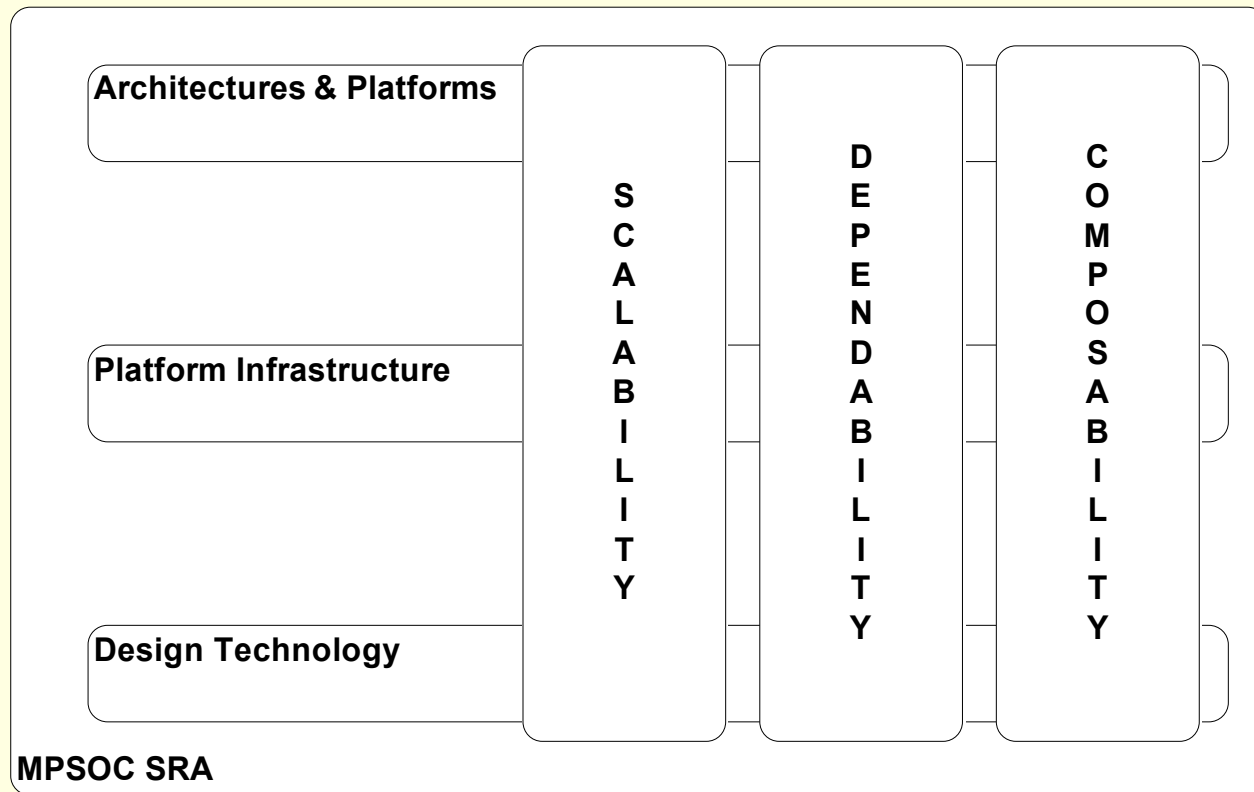
MPSOC Priorities - Dependability

- Managing a system build of heterogeneous HW modules, running several OS and with a very low degree of correlation highly increases the complexity of the process and the design of related tools.
- **“Back in time”** capabilities will **not** be **possible** anymore, as dependencies will have a highly non-deterministic characteristic.
- Architectural features must be **available early** to verification tools, such that specific verification during design development can be performed on architecture instances.
- **Self-test environments** → a necessity for run-time verification.
 - Platform layers to carry information in either direction.
 - Component interaction will be supervised and appropriately (dynamically) directed such that failures are solved at run-time.
 - Provisions for such behavior will impact on performance metrics, therefore suitable design methodologies must be able to provide an as much as possible independent treatment for functional and supervisory mechanisms.

MPSOC Priorities - Scalability

- The accommodation of a large number of processing cores, **possibly not known** at design time
- **3D**: logic, communication architectures and memory hierarchies will extend on a third dimension.
- The increased number of processors with a probable **different provenience** and running **a variety of OS** will raise additional connectivity issues.
- **Design methodologies:**
 - Increase HW platform transparency
 - Fewer dependencies to actual OS procedural catalogues.
 - Larger stress on the compiler technologies.
 - Contract based design

MPSOC Priorities



MPSOC Challenges

- **System adaptability through run-time reconfiguration**

- A means to resolve issues related to fault tolerance in the context of extremely crowded circuitry.
- A way to deal with new requirements, where on-chip resources may be dynamically requested for immediate deployment, replacing current settings.
- A way to (dynamically) improve performance metrics, considering restrictions like device area and power consumption.

- **Performance and resource management**

- Time, power consumption, size, thermal capabilities, application specific quality of service, etc.

MPSOC Challenges

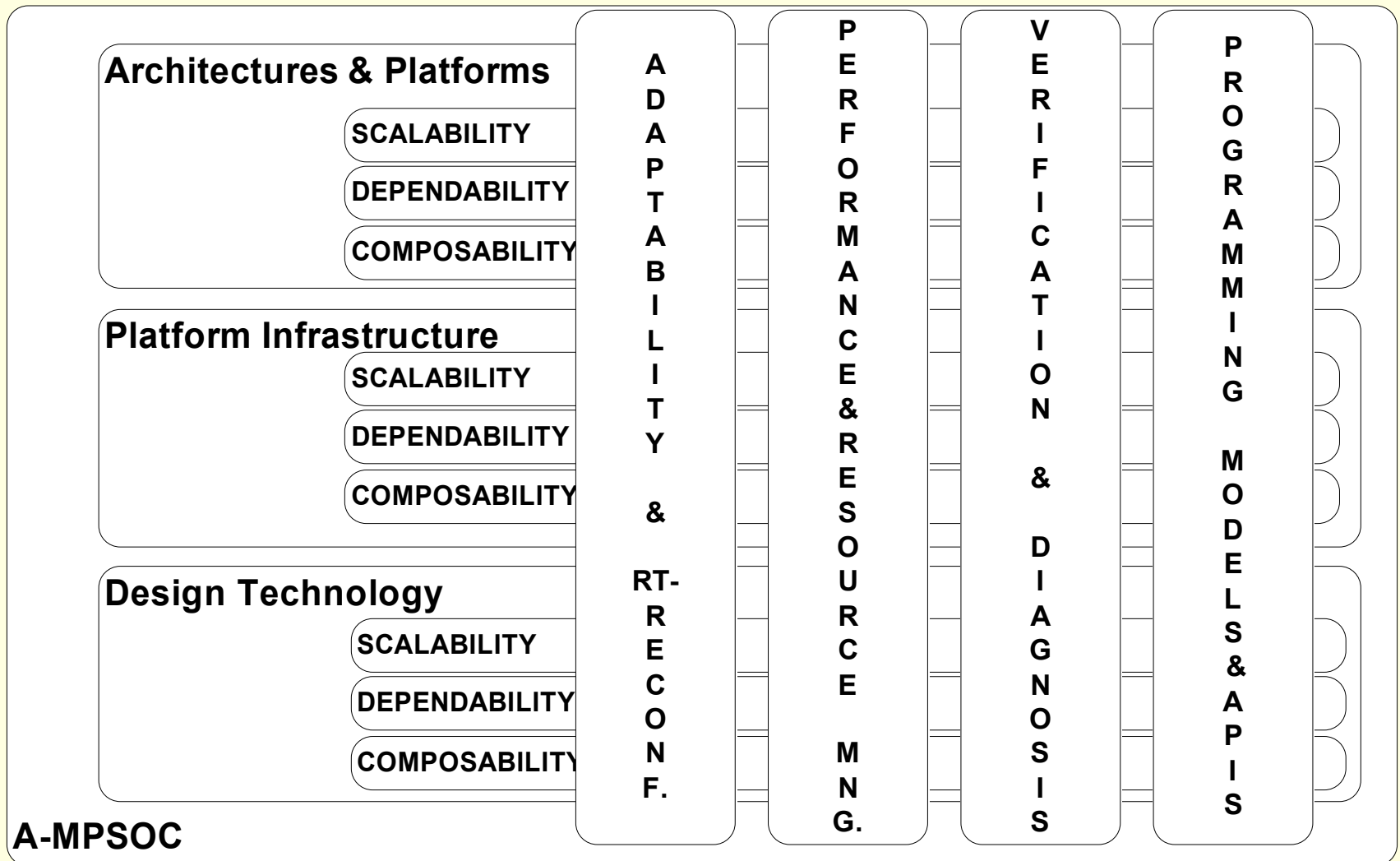
- **Verification & Diagnosis**

- Correct by construction
- Compatibility
- Debugging
- Early assessment

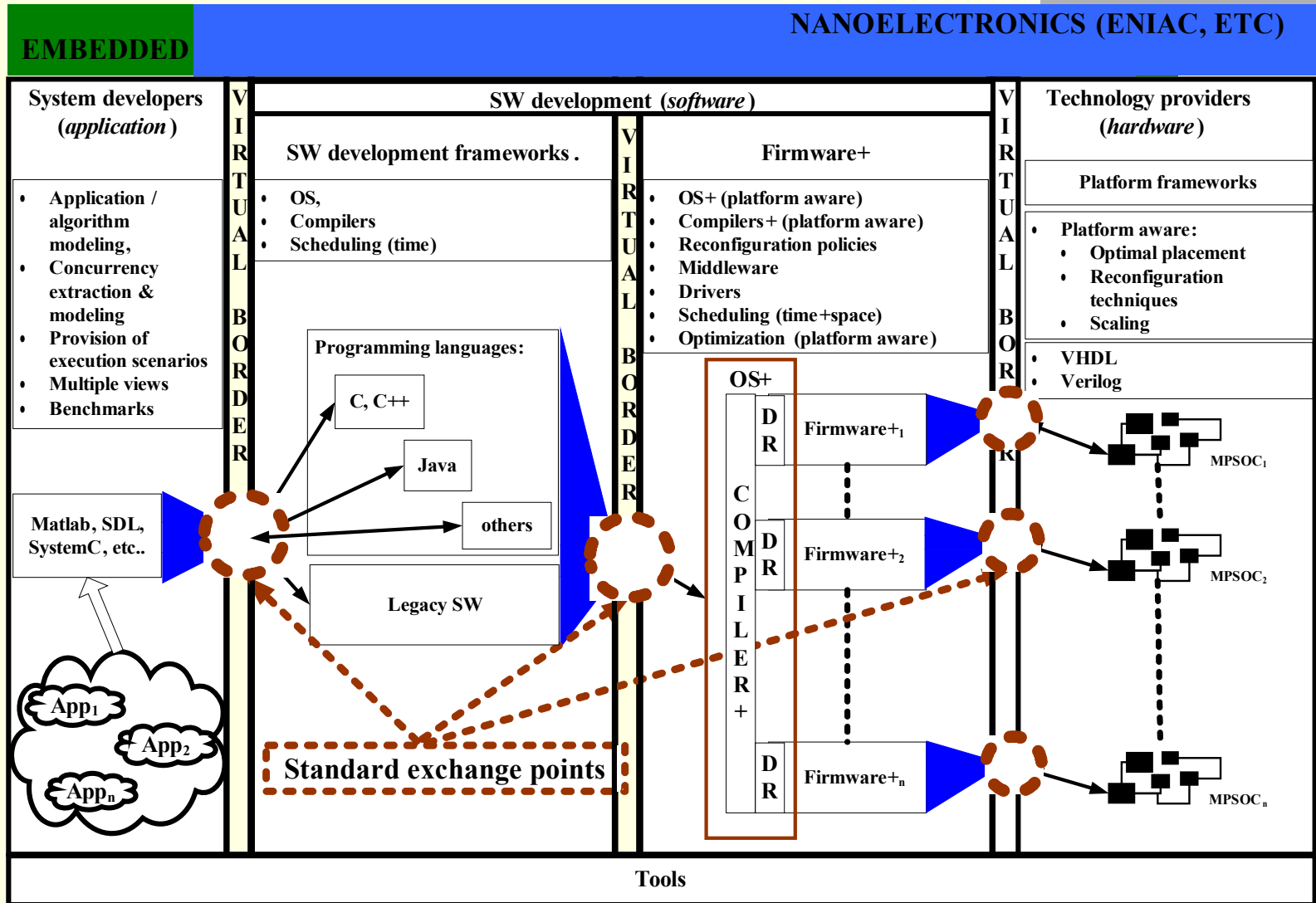
- **Programming models & APIs**

- New parallel programming paradigms
- Memory hierarchies
- HW and SW threads
- Legacy code

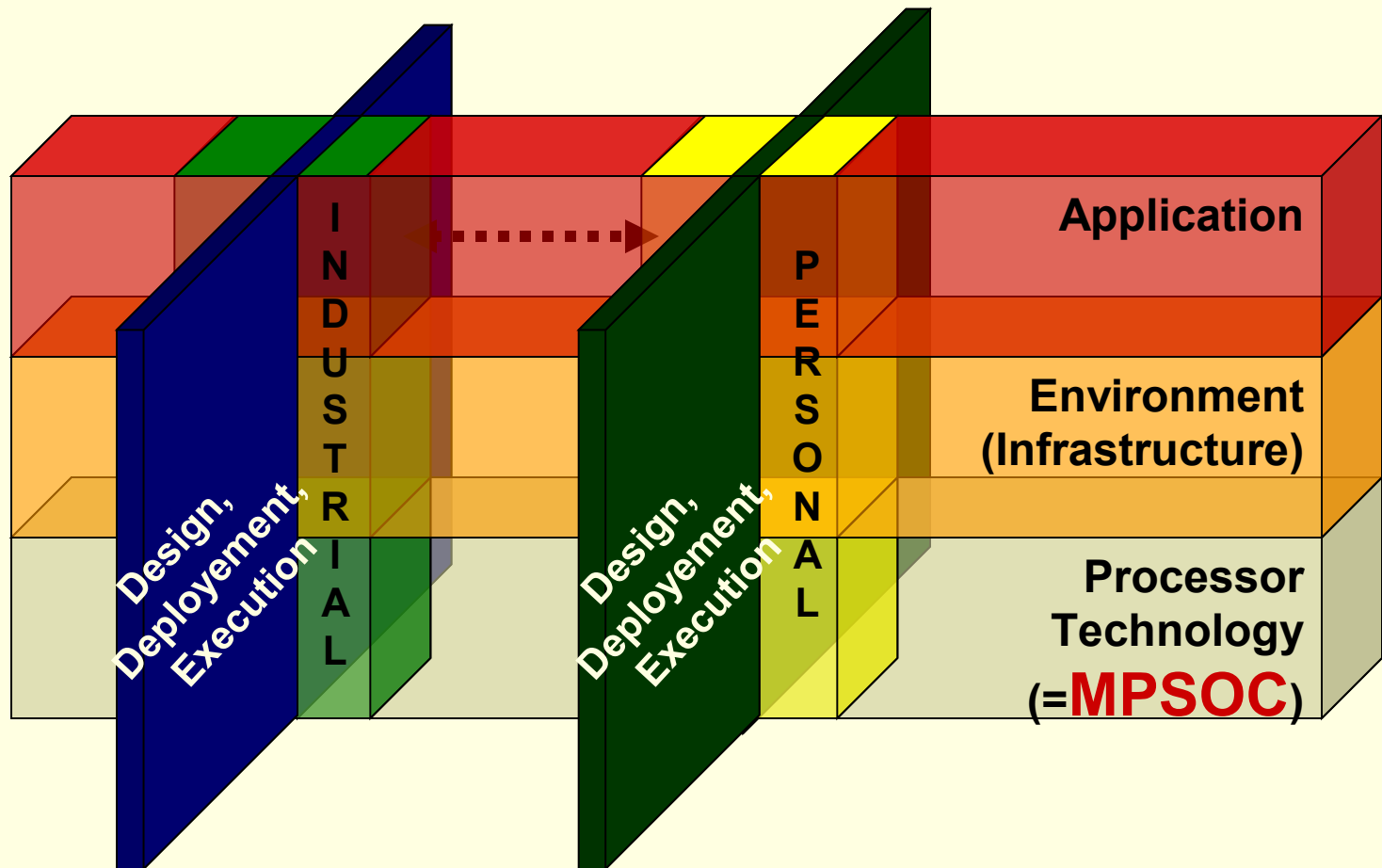
MPSOC Challenges



A consequence...



System Perspectives



Further...

- Web page

(<http://users.utu.fi/tibsec/a-mpsoc/>)

Thank you !