Hierarchical scheduling with Ada 2005

José A. Pulido*, Santiago Urueña*,
Juan Zamorano*, Tullio Vardanega**,
Juan A. de la Puente*

* Universidad Politécnica de Madrid (UPM)

** Università di Padova

Introduction

dit
UPM

- High-Integrity Systems (HIS) face stringent safety and/or security requirements.
- Fully predictable behaviour is mandatory.
  - Large fractions of the Verification & Validation processes have to be implemented.
- Several domain-specific certification standards exist
  - DO-178B (civil avionics)
  - IEC 601-4 (medical systems)
  - IEC 880 (nuclear power plants)
  - ...

HIS & Certification

dit
UPM

- Certification-oriented standards define a range of criticality levels.
  - Whole applications (or parts thereof) are mapped to given levels accordingly
- Complex systems are often composed of applications classified at more than one criticality level.
  - Not all requirements apply to all parts of the application.
- A recurring requirement is to isolate the most critical applications from the less critical ones, so that a failure in the latter does not affect the vital parts of the system.

Providing isolation

dit UPM

- Federated approach
  - Distinct part of one and the same application allocated to distinct computing nodes.
- Integrated architecture
  - The computing platform is divided into a number of *logical partitions*, providing
    - » Temporal isolation
      - A partition cannot exceed a given CPU-time budget.
      - Highly-critical applications will not be prevented from meeting their temporal requirements by misbehaviour of lower criticality applications.
    - » Spatial isolation
      - A partition cannot access memory space outside its allocated storage space, thereby avoiding data corruption by effect of misbehaviours.

Hierarchical scheduling

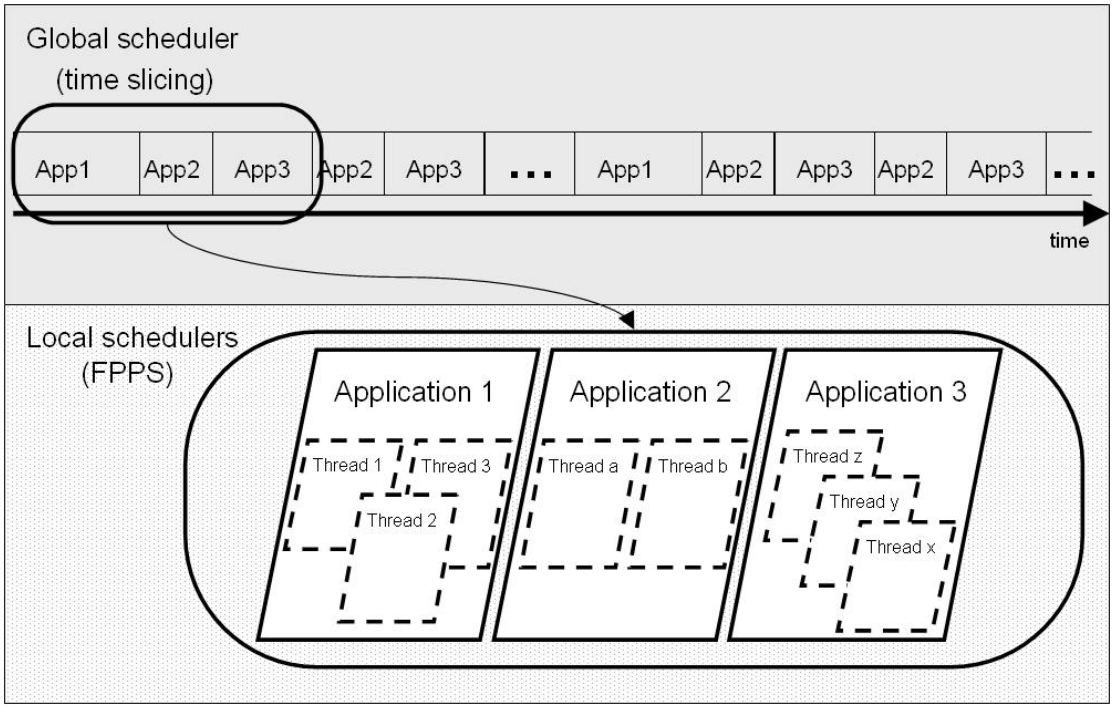
dit UPM

- The key idea behind the concept of *Hierarchical Scheduling* is to use two levels of schedulers
  - A **global scheduler** allocates the processor to a partition (*active partition*).
    - » There is single global scheduler in a computer node
  - A **local scheduler** elects a task to run among the ready tasks in the active partition.
    - » There is a local scheduler for every single partition.
- Several schemes are possible for global and local scheduling.

Fully partitioned architecture

dit UPM

- Completely separated global scheduler and as many local scheduler as partitions.
- ARINC 653 is a well-known example of this approach



Advantages and disadvantages

dit
UPM

- **Advantages**
  - Timeliness and predictability.
- **Disadvantages**
  - Inflexible architecture.
  - Difficult to reconfigure.
  - Rigid communication scheme.
  - Hard to accommodate sporadic tasks effectively.

Server-based architecture

dit UPM

- The global scheduler is supplemented by a set of servers.
- There exist various kinds of servers
  - **Periodic server**
    - » Fixed period.
    - » Executes its tasks until the capacity is exhausted.
    - » Unused capacity is lost.
  - **Deferrable server**
    - » Similar to the periodic server.
    - » Unused capacity is kept for the rest of the period.
  - **Sporadic server**
    - » Capacity is replenished *some time* after it has been used.
    - » More complex rules.

Advantages and disadvantages

dit UPM

- **Advantages**
  - Improves the flexibility of the system.
  - High level of timeliness and predictability.
- **Disadvantages**
  - Complex implementation.

Ada 2005

dit UPM

- Enhancements in the Ada 2005 standard enable an attractive new approach to hierarchical scheduling.
  - New scheduling policies (EDF, RR, non-preemptive FIFO).
    - » Priority-inversion control with SRP.
  - Mixed scheduling policies for priority bands.
  - Execution-time monitoring mechanisms.
    - » Clocks, timers and group budgets.

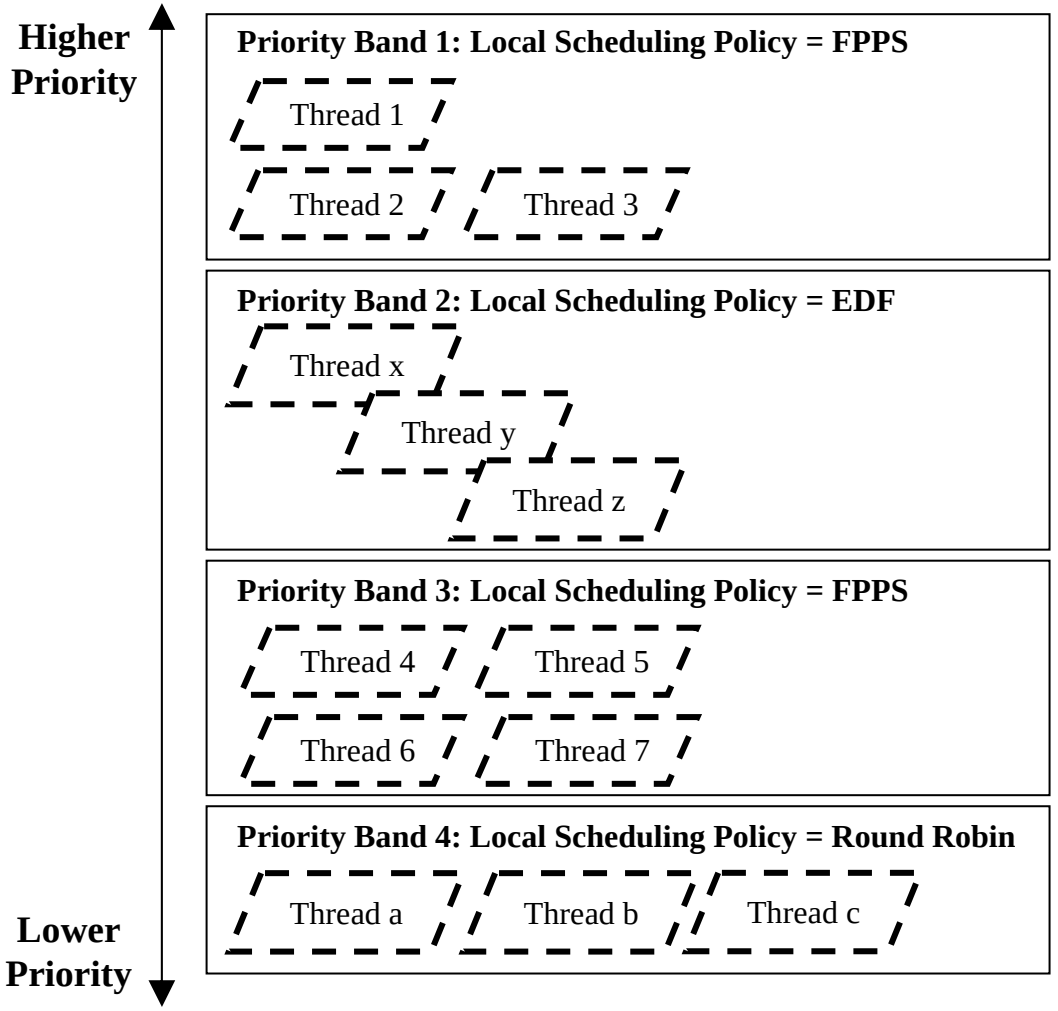
Priority Bands

dit UPM

- Logical partitions mapped to groups of tasks within separate priority bands.
  - The global and local schedulers are integrated into a single framework.
  - The global scheduling policy is still based on Fixed Priority Pre-emptive Scheduling (FPPS).
  - (Possibly) different task dispatching policies for each partition (= task group & priority band) make up the secondary schedulers.

Priority band architecture

dit
UPM



Temporal isolation

dit UPM

- Temporal isolation between partitions.
  - Design time
    - » Priority assignments + temporal analysis.
  - Run-time
    - » Execution-time monitoring mechanisms.
      - Group budgets to detect overloads at partition level.
      - Execution-time timers to detect overloads at the task level.
      - Delay until to next minimum inter-arrival time to enforce minimum separation between successive activations of sporadic tasks.

Mapping rules

dit UPM

- Allocate a priority band to every logical partition.
- Allocate a group budget to every partition
  - With all the tasks added to the group.
- Assign priorities to the partition directly linked to their criticality levels.
  - Non optimal as DMS, etc…
  - If an overload occurs, low-critical tasks will miss their deadlines before than the high critical ones.
- Communications
  - Intra-partition: protected objects as usual.
  - Inter-partition: protected objects with the ceiling priority in the band of the highest priority partition.

Pending questions

dit
UPM

- Compliance with the Ravenscar profile.
  - Execution-time timers and group budgets currently excluded from the RP.
  - Range of recovery actions when an overload is detected is limited.
- Dynamic semantics
  - What if a timer expires inside a protected object?
    - » The LRM compels us to wait until the task leaves the PO.

Spatial isolation

dit
UPM

- Design time.
  - Rely on compiler and on static analysis techniques.
    - » E.g. SPARK tools.
- Run time
  - HW-based memory protection mechanisms.
  - Some processors have only elementary protection.
    - » E.g. fence registers.

Implementation

dit
UPM

- Pilot implementation under way
  - Extended real-time kernel
    - » Based on ORK and AdaCore developments.
    - » “Extended RP”.
    - » Expected by next year.
  - GNAT for Leon
    - » Based on GNAT for ERC32 and new GNAT version.
    - » Expected by Q4 2006.

Conclusions

dit
UPM

- Industrial partners demand a new approach to HIS with different criticality levels.
- ARINC-653 affords a good level of safety but it is highly inflexible.
- The server-based approach is an interesting approach that goes on the right direction.
- The priority bands approach, based on the new features introduces by the Ada 2005 standard, is a flexible, portable and comparatively simple way to implement temporal isolation.

Future work

dit
UPM

- Complete pilot implementation.
- Response-time analysis.
- Deal with spatial isolation.
- Extend to distributed systems.