Resource Reservation & Resource Servers
Design of Real-Time Systems
(the **Autosar** approach in automotive industry)
Design of Real-Time Systems
(the Autosar approach in automotive industry)

Resource Servers

Function Architecture

Hardware Architecture

Physical Architecture

CPU1

CPU2

CPU3

Phy-node 1

Phy-node 2
Resource Reservation

**Application**
Hard real-time, Soft real-time, Others

**Platform**
Hardware Resources: CPU cycles, memory blocks ...
Resource Reservation/Virtual Resources

**Application**
Software Components: Hard real-time, Software real-time, Others

Resource Servers:
e.g. HRT servers, SRT servers, ...

Platform
Hardware Resources: CPU cycles, memory blocks ...
Resource reservation by “servers”

- **periodic server** $T_s=(C_s, P_s)$ for processing aperiodic workload and a queue to buffer tasks that arrive at different times

- Resource servers: examples
  - Polling Server (bandwidth non-preserving)
  - Deferrable Server (bandwidth preserving)
  - Priority Exchange Server (bandwidth preserving)
Polling Server (PS)

• **Idea:**
  – Create a periodic task (a server) with period $Ts$ and capacity $Cs$ (the allowed computing time in each period – allocated resource budget)
  – Schedule the server as a periodic task ($Cs$, $Ts$)

• **Run time behaviour:**
  – Once the server is active, it serves all pending (buffered) aperiodic requests within its capacity $Cs$ according to other algorithms e.g FCFS, SJF etc
  – If no aperiodic requests, the capacity is lost: if a request arrives after the server has been suspended, it must wait until the next polling period
Deferrable server (PS preserving capacity) [Lehoczky and Sha et al, 87,95]

• It is similar to Polling server
• The only difference is that the capacity of DS will be preserved if no pending requests upon the activation of the server. The capacity is maintained until the end of the server
  – within the period, an aperiodic request will be served; thus improving average response time
Priority Exchange (interesting!)

- Similar to PS and DS, PE has a periodic server (usually with high priority) for serving aperiodic tasks. The difference is in the way how the capacity of the server is preserved.

- **Run Time Behaviour:**
  - If the PE server is currently the task with highest priority but there is no aperiodic request pending, then
    - the periodic task with next highest priority runs and
    - the server is assigned with the periodic task’s lower priority
  - Thus the capacity of the server is not lost but preserved with a lower priority (the exchange continues until new aperiodic requests arrive)
Implementation of Resource Servers

• Periodic Servers
  – Servers as periodic tasks:
    • Hart Real-Time: implicit/constrained deadline
    • Soft Real-Time: arbitrary deadline/non-deadline

• Other servers
  – E.g. Bounded Delay Server
    • Round Robin + Worst Case Delay
Resource server: Example

Static server
\{[a_i, b_i]\}

Periodic server
(P, C)

Explicit Deadline Periodic : (P, C, D)
Suply Bound Function (sbf)
(abstract model of resource servers)

Over $L$, you are guaranteed at least $sbf(L)$ cycles.
The SBF of a "perfect" processor

Over L "cycles", you are guaranteed at least L "cycles"
Linear Lower Bound of SBF

Over L, you are guaranteed at least $sbf(L)$ cycles
Bounded Delay Server: \((r,d)\)

Over \(L\), you are guaranteed at least \(\text{sbf}(L)\) cycles.
Workload models

- Non-periodic tasks
- Periodic tasks
- Sporadic tasks

- Demand bound function: $dbf(t)$
  - over time interval $t$, the accumulated amount of work to be computed
Example: Periodic task

<table>
<thead>
<tr>
<th>Length of intervals</th>
<th>#CPU cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
</tr>
</tbody>
</table>
Schedulability analysis: \( \text{for all } t, \text{ dbf}(t) \leq \text{ sbf}(t) \)
Schedulability

#CPU cycles

length of intervals

R: resource supply

W: workload
Resource Reservation/Virtual Resources

Application
Software Components: Hard real-time, Software real-time, Others

Application Scheduling

Resource Servers:
e.g. HRT servers, SRT servers, Xservers for house keeping, ...

Rate monotonic scheduling

Platform
Hardware Resources: CPU cycles, memory blocks ...
Application Scheduling

• Given
  – a group of applications e.g. HRT apps,
  – a Resource Server

• Design a Scheduler s.t.
  – no deadline miss
Application Schedulers

• Non Real-Time Applications
  – Schedule to run according to the budgets allocated by the servers e.g. “60% resource rate”

• Real-Time Application: (W, R, A)
  – W: Workload model e.g. dbf
  – R: Resource server e.g. sbf
  – A: Scheduling algorithm e.g., EDF

• Research Questions
  – Schedulability analysis
    • dbf(t) <= sbf(t) for all t (Necessary condition for all A’s)
  – Synthesis problems e.g.
    • Given W, R, find A s.t. W is schedulable
    • Given W, A, find the “minimal” R s.t. W is schedulable.
    • Given W, R, A: calculate R*: the remaining resource after W
Hierarchical Application Scheduling

- **Given**
  - a collection of applications: workloads
  - a set of Servers
- **Decompose**
  - the apps into “sub-apps”
  - Servers into “sub-servers”
Resource Virtualization (Buttazo et al)

Fig. 1. Architecture overview.