SAPM
Performance
Performance General Scenario

<table>
<thead>
<tr>
<th>Portion of Scenario</th>
<th>Possible Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>Internal or external to the system</td>
</tr>
<tr>
<td>Stimulus</td>
<td>Arrival of a periodic, sporadic, or stochastic event</td>
</tr>
<tr>
<td>Artifact</td>
<td>System or one or more components in the system</td>
</tr>
<tr>
<td>Environment</td>
<td>Operational mode: normal, emergency, peak load, overload</td>
</tr>
<tr>
<td>Response</td>
<td>Process events, change level of service</td>
</tr>
<tr>
<td>Response Measure</td>
<td>Latency, deadline, throughput, jitter, miss rate</td>
</tr>
</tbody>
</table>
Making the Scenario Specific

• We need to say something about the distribution of the arrival of the stimuli
  – E.g. The inter-arrival time is always greater than 1.0 secs
  – How is this different from the arrival rate is less than 1 per second?

• Any stimulus needs to be processed within 2 seconds of arriving.

• The responses should appear in the same order as the stimuli
A possible architecture

queue → process → output
Be specific about the architecture

• We need to say something about the capacity of the processor:
  – The worst case processing time for a stimulus is 1.5 seconds best case time is 1.0 secs
  – The processor can only process one stimulus at a time.
• We need to say that the queue capacity is 7 stimuli (or some other).
• This architecture fails the scenario (why?)
Performance Tactics

- Control Resource Demand
  - Manage Sampling Rate
  - Limit Event Response
  - Prioritize Events
  - Reduce Overhead
  - Bound Execution Times
  - Increase Resource Efficiency

- Manage Resources
  - Increase Resources
  - Introduce Concurrency
  - Maintain Multiple Copies of Computations
  - Maintain Multiple Copies of Data
  - Bound Queue Sizes
  - Schedule Resources
A possible architecture

queue

process

process

output
New Architecture

• This passes the scenario – why?
• Suppose the processing time for the stimulus was much more variable (e.g. 0.2 secs to 1.5 secs) – does the architecture still satisfy the scenario?
Control Resource Demand Tactics

• **Manage the sampling rate** (not always applicable) – ensure you do not have too much to handle.

• **Limit the event response** – if you are receiving too many events, throw some away.

• **Prioritize events** – some need a response in a certain time – some don’t

• **Reduce overhead** – can you take resource out of handling an event?

• **Improve the efficiency of processing** – so you can handle more with the same processing
Manage Resources

• Increase resources
• Introduce concurrency
• Maintain multiple copies of compute and/or data
• Bound queue sizes
• Schedule resource when there is contention (hard scheduling for highest priority events)
Checklist: Allocation of Responsibilities

• Work out areas responsibility of that require heavy resource use to ensure time-critical events take place.
• Work out processing requirements.
• Take account of:
  – Responsibilities arising from threads crossing boundaries of responsibility
  – Responsibilities for thread management
  – Responsibilities for scheduling shared resources
Checklist: Coordination Model

• What needs to coordinate.
• Is there concurrency? Ensure it is safe.
• Ensure coordination is appropriate for the style of stimulus.
• Ensure the properties of the coordination model are good for the stimuli and concurrency control?
Checklist: Data Model

• Determine what parts of the data model will be heavily loaded or have tight time constraints.

• Then:
  – Would keeping multiple copies help?
  – Would partitioning the data help?
  – Is it possible to reduce processing requirements for the data?
  – Does adding resource help deal with data bottlenecks?
Checklist: Mapping Among Architecture Elements

• Does colocation of some components reduce latencies?
• Ensure components with high processing needs are allocated to big processors
• Consider introducing concurrency when you map.
• Consider whether some mappings introduce bottlenecks (e.g. allocating non-interfering tasks to the same thread)
Checklist: Resource Management

• Work out what needs high levels of resource
• Ensure these are monitored and managed under all operating modes.
• For example:
  – Time critical components
  – Thread management
  – Prioritization
  – Locking and scheduling strategies
  – Deploying additional resource to meet elevated load.
Checklist: Binding time

• Look at when you bind.
• Consider the cost of binding at different times
• Try to avoid performance penalties caused by late binding.
Checklist: Choice of Technology

• Is the technology right to let you meet hard deadlines and resource use (e.g. use a real-time OS with proper scheduling).

• You need:
  – Good scheduling
  – Priorities
  – Policies for demand reduction
  – Allocating processing to tasks
  – Other performance-related measurement and management.
Summary

• For performance you need to ensure resource is effectively monitored and managed.
• Architecture gives you a good level to do this.
• Next we consider Security.