UNCLASSIFIED

Australian Government
Department of Defence
Defence Science and
Technology Organisation

UniSA

# Considerations for a Framework for Specification and Measurement of Reserve Capacity in Software-Intensive Systems

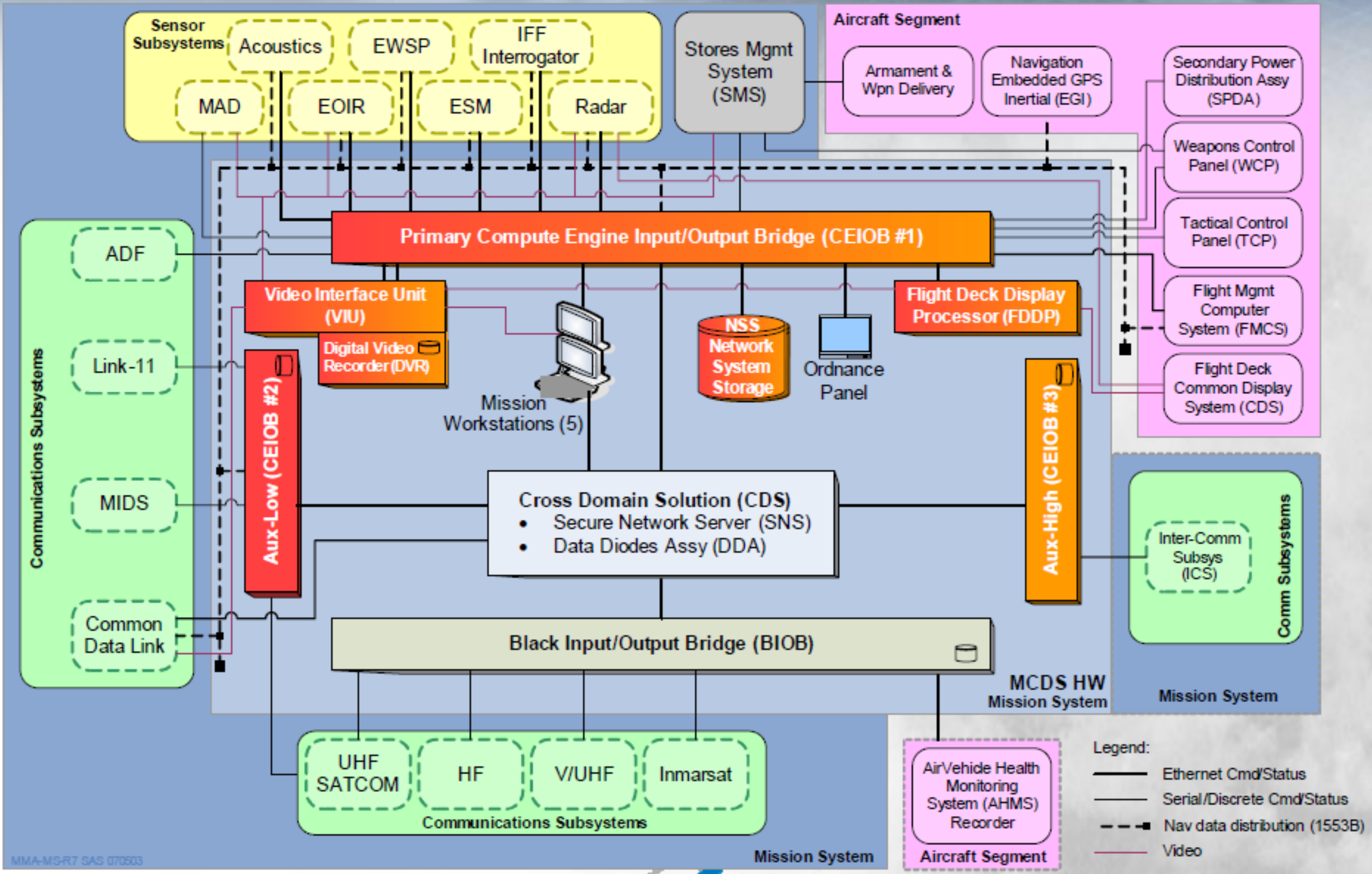
Trent O'Connor
Stephen Cook

DSTO Science and Technology for Safeguarding Australia

*How can you write meaningful requirements for reserve capacity of a software intensive system?*

NAVY
951

# P-8A Mission System Architecture & Interfaces


Sensor Subsystems
Acoustics
EWSP
IFF Interrogator
MAD
EOIR
ESM
Radar
Stores Mgmt System (SMS)
Aircraft Segment
Armament & Wpn Delivery
Navigation Embedded GPS Inertial (EGI)
Secondary Power Distribution Assy (SPDA)
Weapons Control Panel (WCP)
Tactical Control Panel (TCP)
Flight Mgmt Computer System (FMCS)
Flight Deck Common Display System (CDS)
Communications Subsystems
ADF
Link-11
MIDS
Common Data Link
Primary Compute Engine Input/Output Bridge (CEIOB #1)
Video Interface Unit (VIU)
Digital Video Recorder (DVR)
Mission Workstations (5)
NSS Network System Storage
Ordnance Panel
Flight Deck Display Processor (FDDP)
Aux-Low (CEIOB #2)
Aux-High (CEIOB #3)
Cross Domain Solution (CDS)
• Secure Network Server (SNS)
• Data Diodes Assy (DDA)
Inter-Comm Subsys (ICS)
Comm Subsystems
Black Input/Output Bridge (BIOB)
MCDS HW
Mission System
Mission System
UHF SATCOM
HF
V/UHF
Inmarsat
Communications Subsystems
Mission System
Air Vehicle Health Monitoring System (AHMS) Recorder
Aircraft Segment
Legend:
Ethernet Cmd/Status
Serial/Discrete Cmd/Status
Nav data distribution (1553B)
Video
MMA-MS-R7 SAS 070503

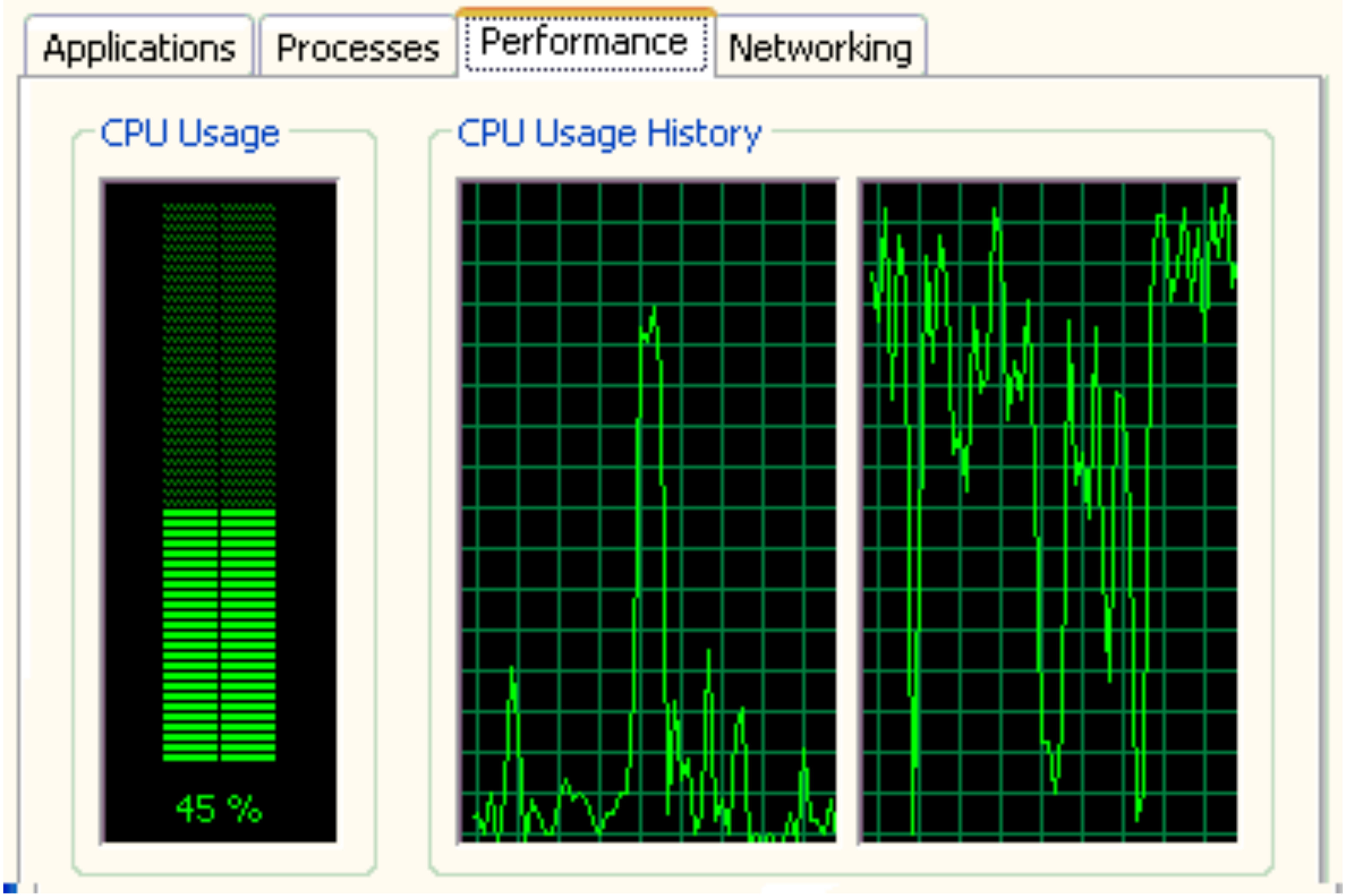
*The [system] shall provide not less than 300% growth capacity for data processing, storage and transport…*

*The [system] shall provide not less than ~~300%~~ 100% growth capacity for data processing, storage and transport…*

*The [system] shall provide not less than ~~300%~~ 100% growth capacity for data processing, storage and transport…*

*… except for embedded COTS/NDI that does not require software upgrade*

*… and some other things*


Applications
Processes
Performance
Networking
CPU Usage
45 %
CPU Usage History

# Observed Problem – Inadequate Reserve Capacity

- Problem - Software-intensive systems delivered with insufficient reserve capacity in data processing resources e.g. CPU, memory, storage or networks.

- Consequences:
  - Poor resilience to mismatches between the test environment and the real world
  - Reduced robustness to load transients
  - Reduced maintainability
  - Poor growth potential

- Changing hardware in embedded environments can cost a lot of time and money.

- Prefer to get adequate reserve capacity at delivery, but this can be hard to describe and verify

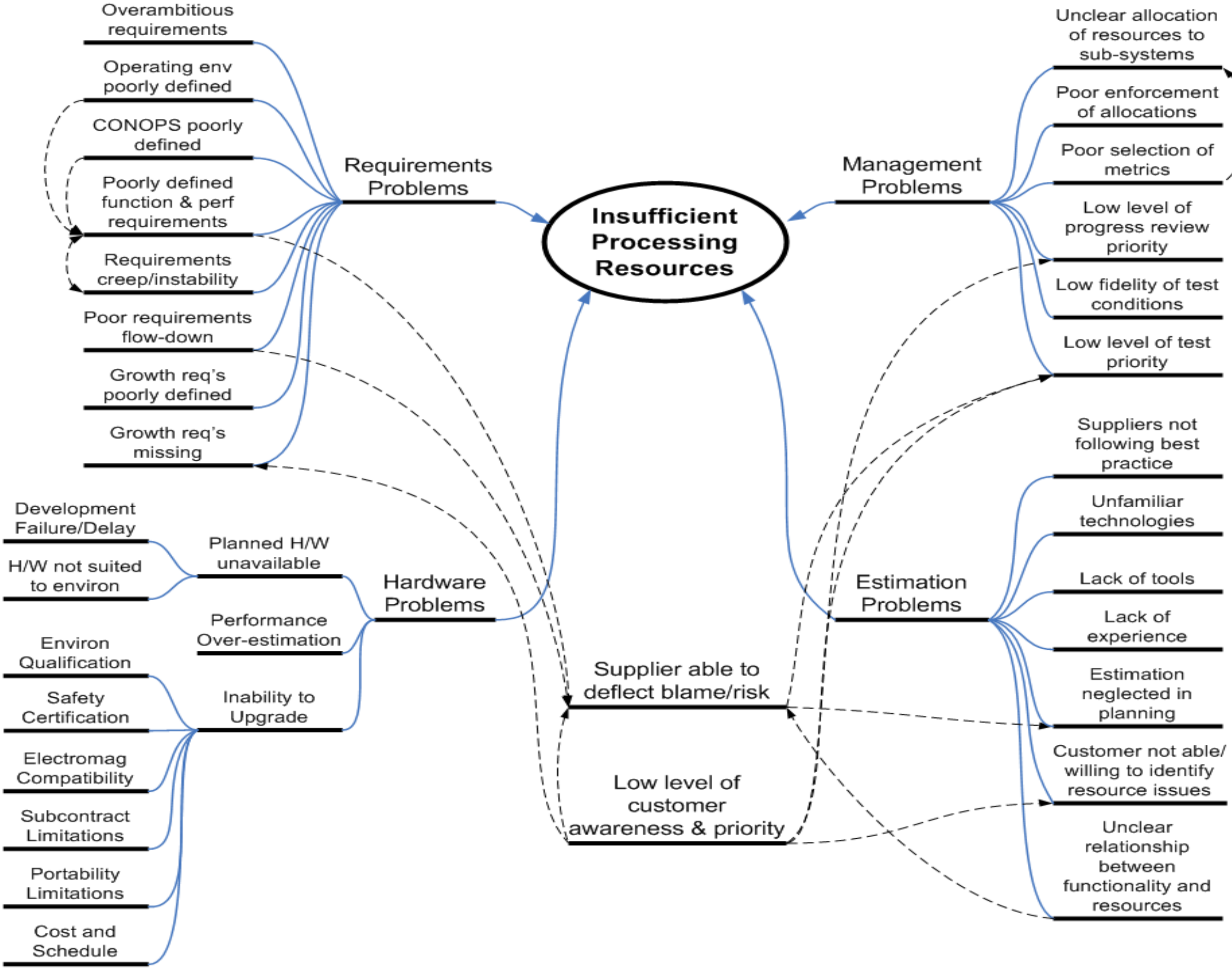
# Existing Practice - Findings

Structured interviews with 13 practitioners:

- Reserve capacity problems generally acknowledged but not documented
- No standardised approach to specification or verification
- Dissatisfaction with contractual CPU utilization requirements
- Requirements developed with only a vague idea of verification
- Design goals for reserve capacity frequently watered down or disregarded as development progresses

Examined specifications and some verification procedures for military software-intensive systems acquisitions:

- Reuse of unsuitable requirements from past projects
- Verification methods not matched to requirements



Insufficient Processing Resources
Requirements Problems
Overambitious requirements
Operating env poorly defined
CONOPS poorly defined
Poorly defined function & perf requirements
Requirements creep/instability
Poor requirements flow-down
Growth req's poorly defined
Growth req's missing
Management Problems
Unclear allocation of resources to sub-systems
Poor enforcement of allocations
Poor selection of metrics
Low level of progress review priority
Low fidelity of test conditions
Low level of test priority
Hardware Problems
Planned H/W unavailable
Development Failure/Delay
H/W not suited to environ
Performance Over-estimation
Inability to Upgrade
Environ Qualification
Safety Certification
Electromag Compatibility
Subcontract Limitations
Portability Limitations
Cost and Schedule
Estimation Problems
Suppliers not following best practice
Unfamiliar technologies
Lack of tools
Lack of experience
Estimation neglected in planning
Customer not able/ willing to identify resource issues
Unclear relationship between functionality and resources
Supplier able to deflect blame/risk
Low level of customer awareness & priority

# Inadequate Reserve Capacity - Causes

- Over-ambitious functional requirements
- Poorly defined functional requirements
- Requirements creep/instability
- Inadequate specification of operating environment
- * Ambiguous or incomplete specification of required growth capacity
- Difficulty of correctly estimating required resources
- Technology and architecture (im)maturity and evolution
- * Unclear allocation of shared resources to sub-functions
- * Lack of understanding and attention by customers
- * Poor usage of metrics
- * Difficulty in tracing responsibility for shortfalls
- Unavailability or underperformance of intended hardware
- Barriers to upgrading hardware to meet processing demands

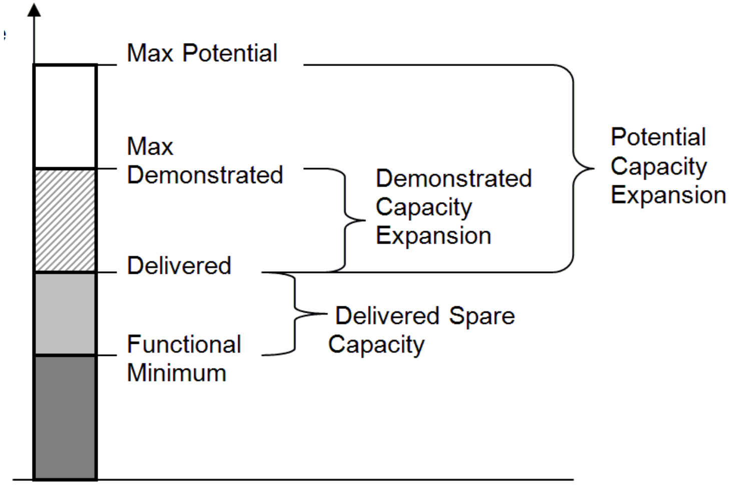
Some of these causes have a common theme, and maybe some common solutions

# Proposal – What is Needed

- A framework of guidance on specifying and measuring reserve capacity

- Could include:
  - Defined terminology for describing the parameters of interest
  - Recommendations or templates of clauses for incorporation into contract wording
  - Recommended approaches for measuring and testing spare capacity and documenting the related conditions and assumptions.

- The paper describes the desirable properties for such a framework



# Framework Attributes – 1

- Domain-independent – Applicable to as broad a range of application domains as possible
- Design-independent – Makes a minimum of assumptions about the hardware or software elements of the solution
- Comprehensive coverage - Covers all relevant resource types
- Scalable – Applicable to a broad range of system sizes and complexities
- Allocable to subsystems – Compatible with functional decomposition of a system

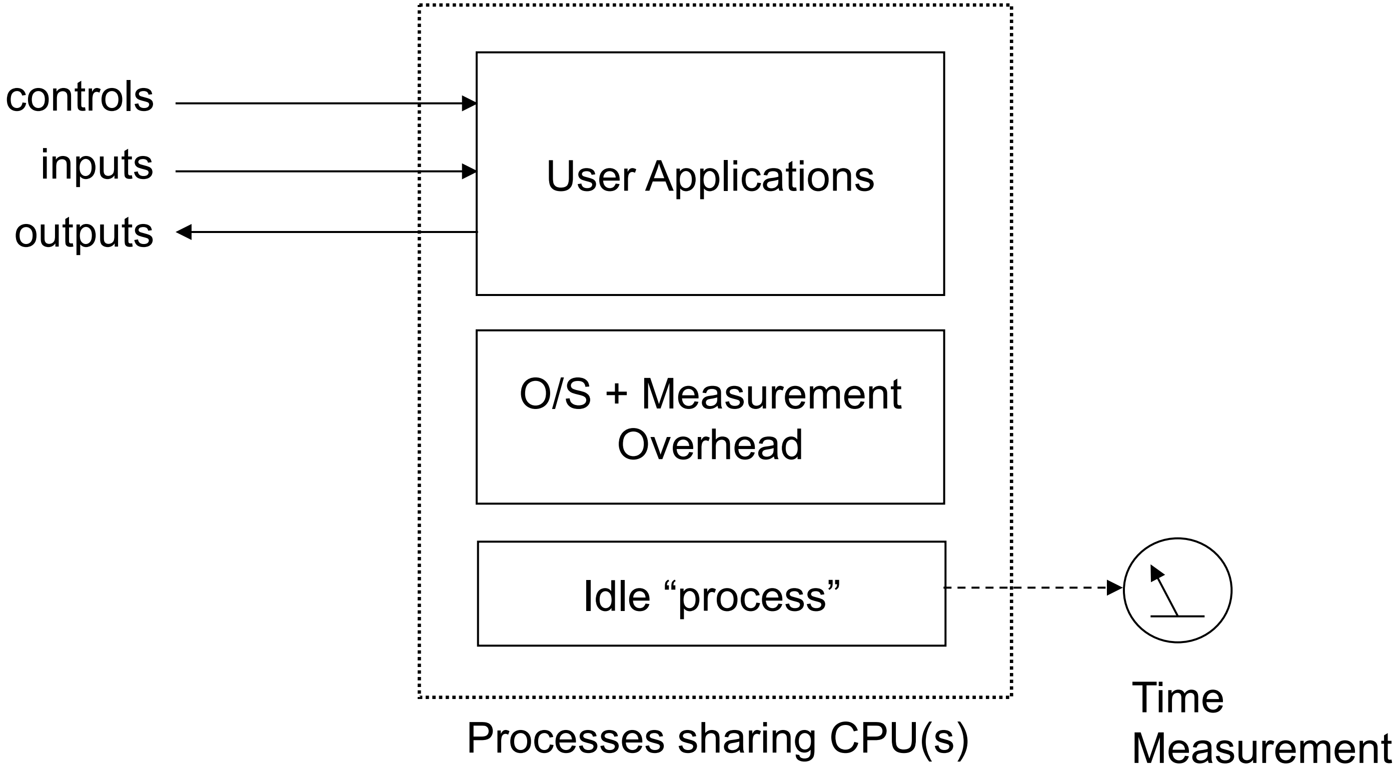
# Framework Attributes – 2

- Tailorable – Provides for waivers or adjustment to the requirements where appropriate
- Flexible in reserve application – Maximises opportunities to use resources rather then have them idle
- Consistent with standards – As consistent as possible with any existing standards that overlap the scope of the framework
- Supports continuous/incremental usage – Can be used throughout development to monitor progress, not only at the end

# Framework Attributes – 3

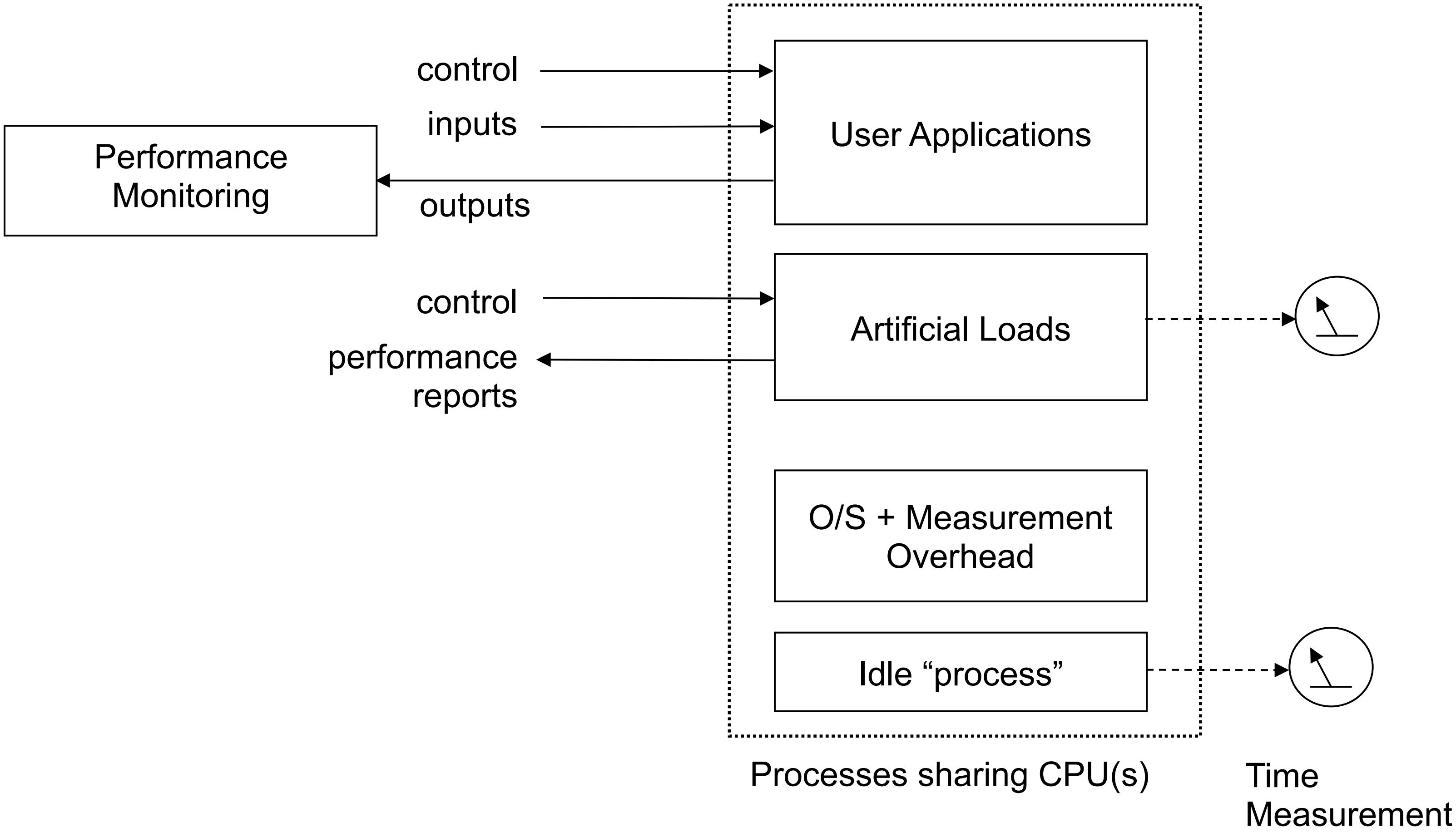
- Generates "good" requirements – Meets the accepted standards for "good" requirements - Correct, Feasible, Complete, Necessary, Prioritised, Unambiguous, Verifiable, Consistent, Modifiable, Traceable, Understandable, Organised, Non-redundant
- Generates "good" metrics – Meets the accepted standards for "good" metrics. Linearity, Reliability, Repeatability, Ease of Measurement, Consistency, Independence
- Feasible – Capable of being put into practice without introducing excessive delay or cost
- Proven – Has been validated and shown to improve program outcomes

# Traditional Utilization Measurement



Is this reliable?
Is this valid for our problem ?

# Measurement with Artificial Loads



Reserve CPU% = Artificial load CPU%? Maybe.

# Conclusions So Far

- Problems with reserve data processing capacity exist and are significant
- There is a lack of tools and guidance to avoid such problems
- Better reserve capacity specification and measurement tools are conceivable, but will need to meet many conditions to obtain widespread acceptance and use

# Next Steps

- Develop valid methods for verifying reserve capacity
- Develop a case that illustrates superior validity over established methods
- Develop requirements aspects that match the verification approaches

# Questions?

… or observations from practice?

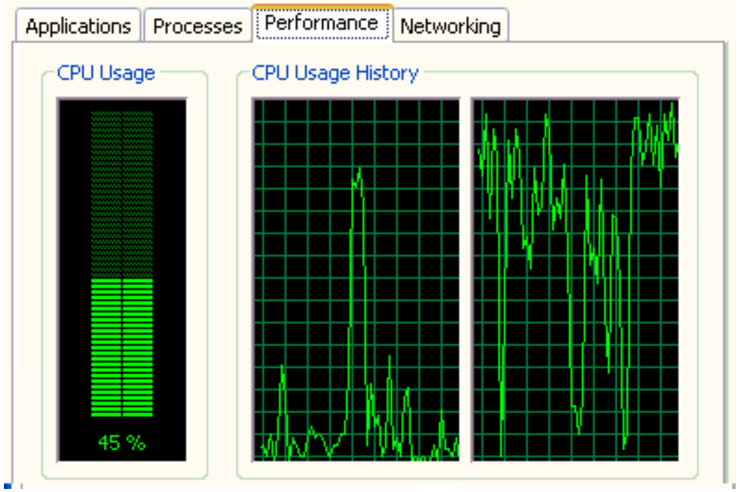
# Backup

# Scoping - When to Apply the Framework

- Customer writes specification, including spare capacity
- Supplier designs system and predicts resource usage
- Supplier builds system and reports resource usage incrementally
- Supplier delivers system and reports resource usage
- Supplier and customer verify resource usage against specification
- Customer monitors resource usage in operational environment

# Poor Specification – Further Example

- *"The [system] shall provide not less than 50 percent reserve capacity in throughput for each system processor, evaluated under worst-case loading conditions."*



- Sounds fair, but ...

- How is "throughput" measured?
- What is the boundary of a "system processor"?
- How is the "worst case" condition identified?
- Should this apply to non-developmental components in the design? If so, how?

- Is such a requirement verifiable?

- If not, what is its contractual significance?