MISSION SOLUTIONS ENGINEERING

Performance Engineering 101

Eric Landrieu
Software Engineer Leader, MSE

Systems & Software Technology Conference 2011

May 18, 2011

Objective

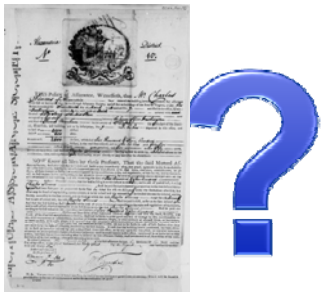
- Provide an understanding of why Performance Engineering is needed
- Introduce Performance Engineering concepts (at a high level)
- Help you understand what your Performance Engineering team does (if you have one)
  - If you do not have one, help you understand what one should be doing for you!

Agenda

- Why do Performance Engineering?
- Questions to ask for Performance Engineering tasks
- Performance Engineering at MSE
- Collecting Performance Data
- The Core Four Resources
- Processes
- Virtualization
- Questions

Why do Performance Engineering?

- Understand system behavior
- Find cause of problems
- Requirements
  - Contracts
  - Specifications
  - Internal Processes

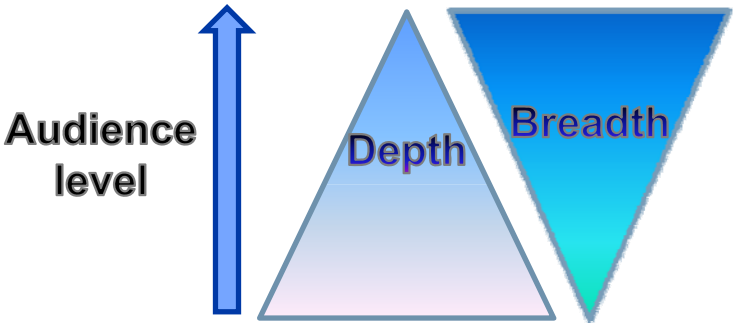


- Provide feedback or information to others
- Predictive Engineering (plan for upgrades, changes)
- Risk Management
  - The quicker a problem is addressed, the lower the cost to address it
- Quality Assurance

Photo source: http://commons.wikimedia.org/wiki/File:Insurance_contract.jpg

Questions to ask when doing Performance Engineering

- **Who** is asking for information?
  - The higher up the information travels, data should contain less detail and be more broad-based



- **What** information is needed?
- **Where** (what systems) do we want to analyze?
- **When** should we analyze?
  - Different needs for short or long timeframes
- **How** should the information be presented?
  - Graphs/tables/text, html/pdf/hardcopy, etc.
- **Why** is the information needed?

Why is the information needed?

- Troubleshooting
  - Immediate data on system/resources where problem(s) occurring
  - Narrow down location of problems to better address them quickly
- Monitoring
  - Realtime data on systems and resources
  - Condensed to highlight possible issues
    - Dashboard view of systems/groups
    - Alerts of present or future issues
- Testing
  - Measure data over period of time on all or subset of system
  - Usually put system under defined load
  - Understand system operation in specific scenarios
- Predictive Engineering
  - Data collected over longer period of time (days, weeks, months, etc.)
  - Analyze trends over time, find future problems, model expected usage

Performance Engineering at MSE

- MSE has been developing software for Aegis for over 40 years
- Performance Engineering is vital in Open Architecture development
- Performed by dedicated experts in Mission Assurance department
  - Independent of individual development organizations
- Performed throughout development lifecycle
- Performed on all baselines
- Standardized sets of tools for system and internal measurement and analysis

Collecting Performance Data

- System and Process data from OS (outside black box)
  - Most modern Operating Systems provide performance data
  - The OS may provide some internal performance data
    - E.g. caching, kernel statistics, data provided to OS by applications
  - OS tools usually provided, but are very raw
  - 3rd Party and Open Source tools available
    - Better manageability
    - Provide intelligence in analyzing the data
  - At MSE, we constantly monitoring for problems or future issues

Photo source: http://commons.wikimedia.org/wiki/File:Digital_spannungspruerfer.JPG

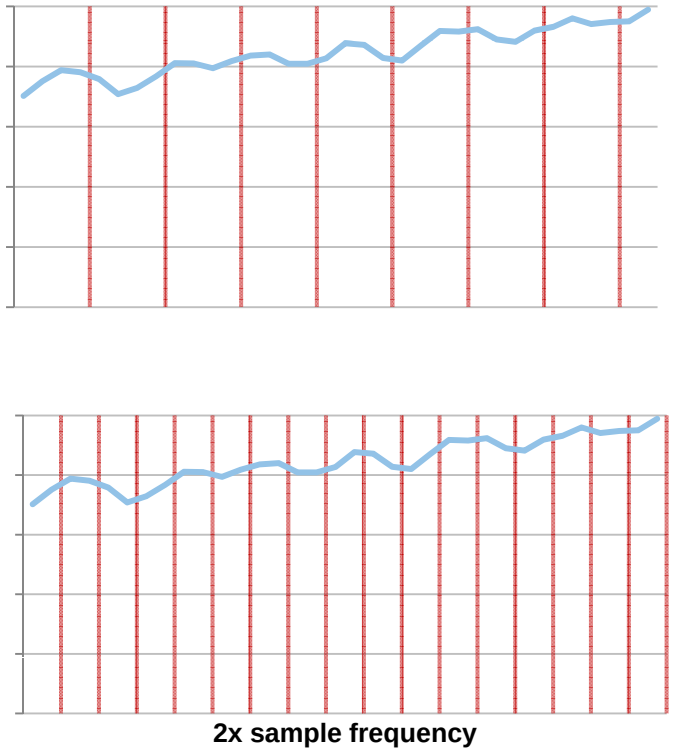
Collecting Performance Data

- Deeper dive into processes (inside the black box)
  - Requires deeper knowledge of architecture
    - Measure timing and resource usage along critical paths in architecture
    - Understand where performance issues are occurring
  - Tools to use depend upon what you are measuring
    - Profilers – code-level and system-level
    - Log files containing timing or resource usage data
    - Internal instrumentation of system
  - Examples
    - Time for system to react to user request (with timings for each step)
    - Timing and sequence of messages to a component
    - Memory utilization in a buffer
    - Profile of CPU Usage by internal component
  - At MSE, we use this for a deeper understanding of performance or troubleshooting data that we see from the system level

Photo source: http://commons.wikimedia.org/wiki/File:Rock_crusher_gears.jpg

Collecting Performance Data

- Collection Intervals
  - How Often do you measure?
  - Higher sample frequency means:
    - Finer level of detail
    - More likely to capture peaks and valleys
    - More data to analyze
    - More effect on system being measured
  - Balance needs and impact to find best collection interval
    - Different needs may necessitate different data intervals
      - Testing and troubleshooting quick-occurring issues may require very high sample rate
      - Predictive engineering often uses longer intervals over a much longer timeframe



2x sample frequency

Collecting Performance Data

- Types of measurements
  - Instantaneous
    - Snapshot of resource at a point in time
  - Delta
    - Total change in resource over time period
      - Often for measuring total usage (e.g. Total Transactions)
    - Change in resource per unit (time) – Rate
      - Measuring against a constraint (e.g. Network bandwidth usage)
  - Time to Complete
    - Often average over a sampling interval
      - Example: Disk response time (time to complete per IO)

Photo sources:
Camera: http://commons.wikimedia.org/wiki/File:Beautycord_camera.jpg
Odometer: http://commons.wikimedia.org/wiki/File:Odometer.jpg
Stopwatch: http://commons.wikimedia.org/wiki/File:Cron%C3%B3grafo_anal%C3%B3gico(REFON-Reynaldo).jpg

Core Four Resources

- The Core Four resources are the primary four that constrain a computer system
  - CPU
  - Memory
  - Disk (storage)
  - Network
- Nearly all performance measurements break down into one or a combination of these
  - Example of combination: Disk swapping activity
    - Caused by a shortage of memory
    - Causes heavy disk activity
    - CPU may have to wait for swapping activity to complete
    - CPU may have to wait for memory to be swapped back from disk

CPU

- Main statistics
  - CPU Utilization (used, idle)
  - CPU Load
  - CPU Ready (queueing and response time)
- Main issues
  - CPU Resource Saturation (all CPUs are near capacity)
  - CPU Core Saturation (one or more, but not all CPUs, near capacity)
    - Unbalanced load, not making best use of resources
  - Confusion in representation
    - Many different units used to represent, what do they all mean?
      - Percentage: % of single CPU, % of total CPU resources
      - CPU Seconds
      - Megahertz (MHz)
      - Jiffies
  - Multiple Cores and Hyperthreading

Photo source: http://commons.wikimedia.org/wiki/File:Sockel7-cpus.JPG

Hyperthreading vs. Multi-core

- Hyperthreading CPU looks to OS like 2 full cores
  - Only 1 core, but two pipelines, can schedule well-balanced (multithreaded) workloads to make more efficient use of the core
- Multicore CPU has more than one actual core on the physical chip
- Multiprocessor systems have two or more physical CPU chips inside
- Systems can have any combination of all three of the above

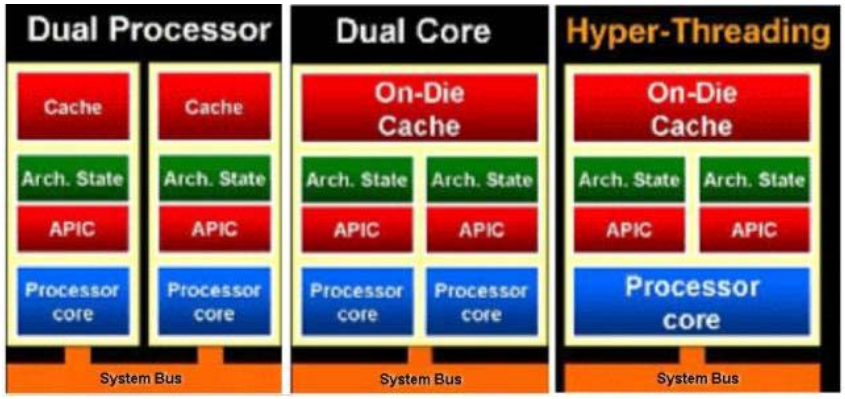


Photo source: http://blogs.msdn.com/b/gauravseth/archive/2006/03/20/555519.aspx

Memory

- Main statistics
  - Memory Used (% or bytes)
  - Memory Free (% or bytes)
  - Swapped Memory
  - Swapping Activity (rate)
- Main issues
  - Memory Overcommitment
    - OS either refuses to give more, or finds ways to get the memory needed
    - Swapping memory to disk is process of last resort
      - Disk access is orders of magnitude slower than memory access
      - The swapping can make other resources (CPU, disk) appear to be the cause of issues

Photo source: http://commons.wikimedia.org/wiki/File:Dimm.jpg

Disk (storage)

- Main statistics
  - IO rates: reads/second, writes/second
  - Throughput rates: bytes read/second, bytes written/second
  - Latencies: seconds per read, seconds per write
  - Disk space: Used, Free
- Main issues
  - Bus saturation
  - Disk throughput saturation
  - Disk space saturation (running out of space)
  - Caching or buffering inefficiencies
- Types of storage
  - Local (hard disk directly connected to system)
  - SAN
  - NAS
- Disk Caching
- Disk Arrays

Photo source: http://commons.wikimedia.org/wiki/File:The_main_Flickr_photo_storage_server.jpg

Network

- Main statistics
  - Packet rates: Received/second and Transmitted/second
  - Throughput rates: Bytes Received/second and Bytes Transmitted/second
- Main issues
  - Port saturation on network device
  - Buffer saturation (send or receive)
- Always remember protocol overhead

Photo sources:
Network card: http://commons.wikimedia.org/wiki/File:Ne1000.jpg
Matryoshka: http://commons.wikimedia.org/wiki/File:Russian-Matroshka_no_bg.jpg

Network Topology

- Different bandwidths and activity levels at different parts of network
- Switching buffers
- Collisions (non-switching networks)
- NIC Teaming and Bonding
- VLANs
- Need to measure throughout network to get full view of where problems may lie

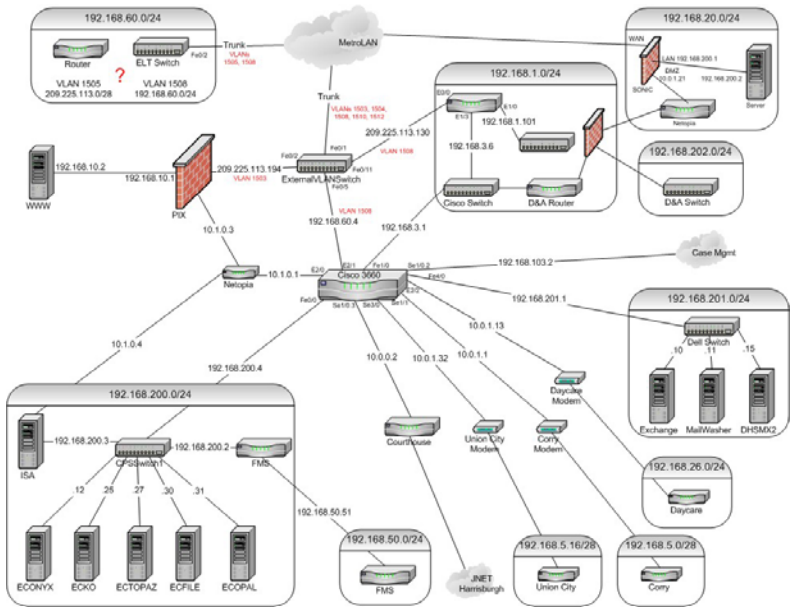


Photo source: http://commons.wikimedia.org/wiki/File:DHS_Network_Topology.jpg

Resource Contention

- Multiple processes running simultaneously on system
  - Sharing finite resources
  - System tries to give each process all resources it wants
- When more requested than resources available, OS has to dole out resources as appropriate to let processes do their work
  - Higher priority processes usually get higher levels of resources

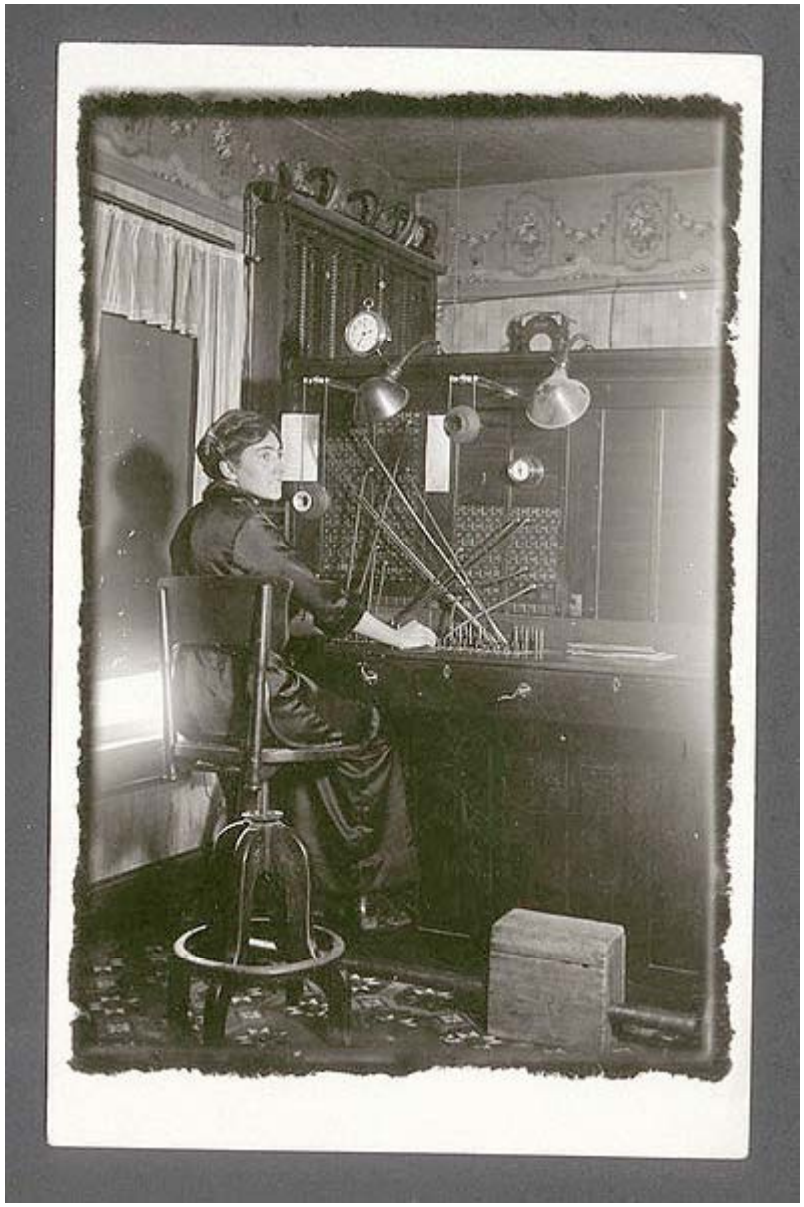


Photo source: http://commons.wikimedia.org/wiki/File:NebraskaHelloGirl.jpg

Processes

- Constrained by the same Core Four as the system
- Often can measure at a basic level from the OS itself
- To go deeper, tools are needed (e.g. profilers)
- Why
  - Narrow focus to source of issues
  - Validate expected operation of processes
  - Measure internal resources and timings to verify proper operation
- Whereas system is “provider” of resources, processes are “consumers” of resources

Photo source: http://commons.wikimedia.org/wiki/File:StateLibQld_1_115664_Feeding_time_for_the_animals,_Blackall_District,_1908.jpg

Virtualization

- Adds a layer of abstraction (complexity)
- Physical vs. Virtual resources
  - Physical: Hardware and hypervisor are provider, virtual machine is consumer
  - Virtual: Virtual Machine is provider, processes in VM are consumer
- Contention occurs for both physical and virtual resources

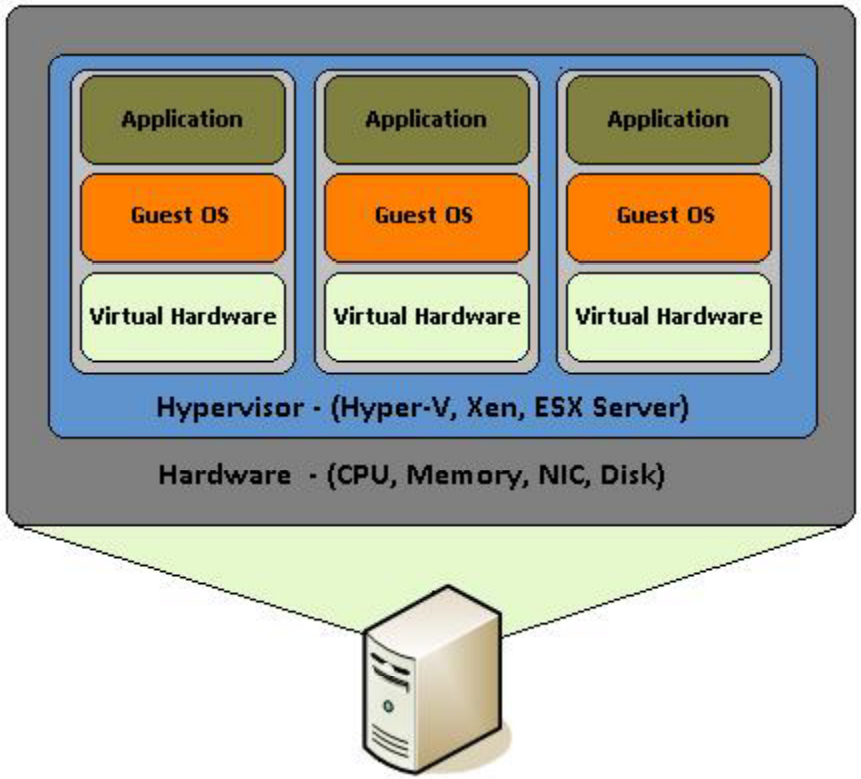


Photo source: http://commons.wikimedia.org/wiki/File:Hardware_Virtualization.JPG

Virtualization

- System-level measurements from within virtual machines may be inaccurate, especially CPU usage
  - Assumptions made that are not accurate in virtual world
  - Recommended whitepaper: http://www.vmware.com/files/pdf/Timekeeping-In-VirtualMachines.pdf
- Storage bandwidth becomes a major factor
  - Virtualization exposes poorly configured storage (SAN)
- Network becomes much more complex
  - Virtual network infrastructure
    - Some may not even connect to physical network
    - Network traffic that does not reach physical network can travel as fast as the host's CPU will allow
- Even if your product does not use virtualization, you may still use it
  - Ideal testing environment

Questions

Acronyms

- CPU – Central Processing Unit
- OS – Operating System
- SAN – Storage Area Network
- NAS – Network-Attached Storage
- VM – Virtual Machine
- NIC – Network Interface Card
- VLAN – Virtual Local Area Network
- IO – Input / Output

MISSION SOLUTIONS ENGINEERING

ERIC LANDRIEU

Software Engineer Leader

t +1.856.252.2135 | eric.landrieu@missionse.com

304 W Route 38 | Moorestown, NJ 08057 | **www.missionse.com**