Fit for Purpose Architecture Selection and Design

Frank J. De Gilio
IBM Corporation

Monday August 12 1:30 pm
Session: 13558

Acknowledgement

- Sally J Eystad – Atlanta
- Eduardo Oliveira - Atlanta
- Don Hemrick - Atlanta
- Satya Sharma – Austin
- Jim Mitchell – Austin
- Joel Tendler – Austin
- William Starke – Austin
- Greg Pfister – Austin
- David Ruth – Austin
- Ndu Emuchay - Austin
- Indulis Bernsteins - Australia
- Ray Williams – Calgary
- Scott D Randall - Cambridge
- Kevin D Dixon – Charlotte
- William Genovese - Charlotte
- Darryl Van Nort – Chicago
- Susan Grabinski – Chicago
- Dennis Robertson - Chicago
- Todd Engen – Columbus
- Susan Schreitmueller – Dallas
- Linda Chidsey - Dallas
- Rick Lebsack – Denver
- Jim Cook - Lexington
- Gord Palin – Markham
- John Schlosser – Milwaukee
- Gary Crupi – Milwaukee
- John Banchy – Minneapolis
- Cindy Banchy - Minneapolis
- Randy Holmes – Minneapolis
- Mickey Sparks – Minneapolis
- John Northway - Minneapolis
- Scott Lundell – Omaha
- DeWayne Hughes - Omaha
- Brian Gagnon – Pittsburgh
- Chris Meurer - Pittsburgh
- Joseph Temple – Poughkeepsie
- Kathy Jackson - Poughkeepsie
- Colleen Ashe - Poughkeepsie
- Pak-kin Mak – Poughkeepsie
- Frank DeGilio – Poughkeepsie
- Hilon Potter – Poughkeepsie
- William Clarke – Poughkeepsie
- Timothy Durniak – Poughkeepsie
- Gururaj Rao - Poughkeepsie
- Jim Barros - Raleigh
- Steve Kunkel – Rochester
- Craig Wilcox – Rochester
- Edith Lueke – Rochester
- Mark Dixon - Sacramento
- Dale Martin - San Francisco
- Jeff Josten - Santa Teresa
- Bill Reeder - Seattle
- Danny Williams – UK
- Jeffrey Callan - Waltham
- Wakaki Thompson – Wayne
- Edward Hoyle - Winston Salem
- James Huston - White Plains

Caveats

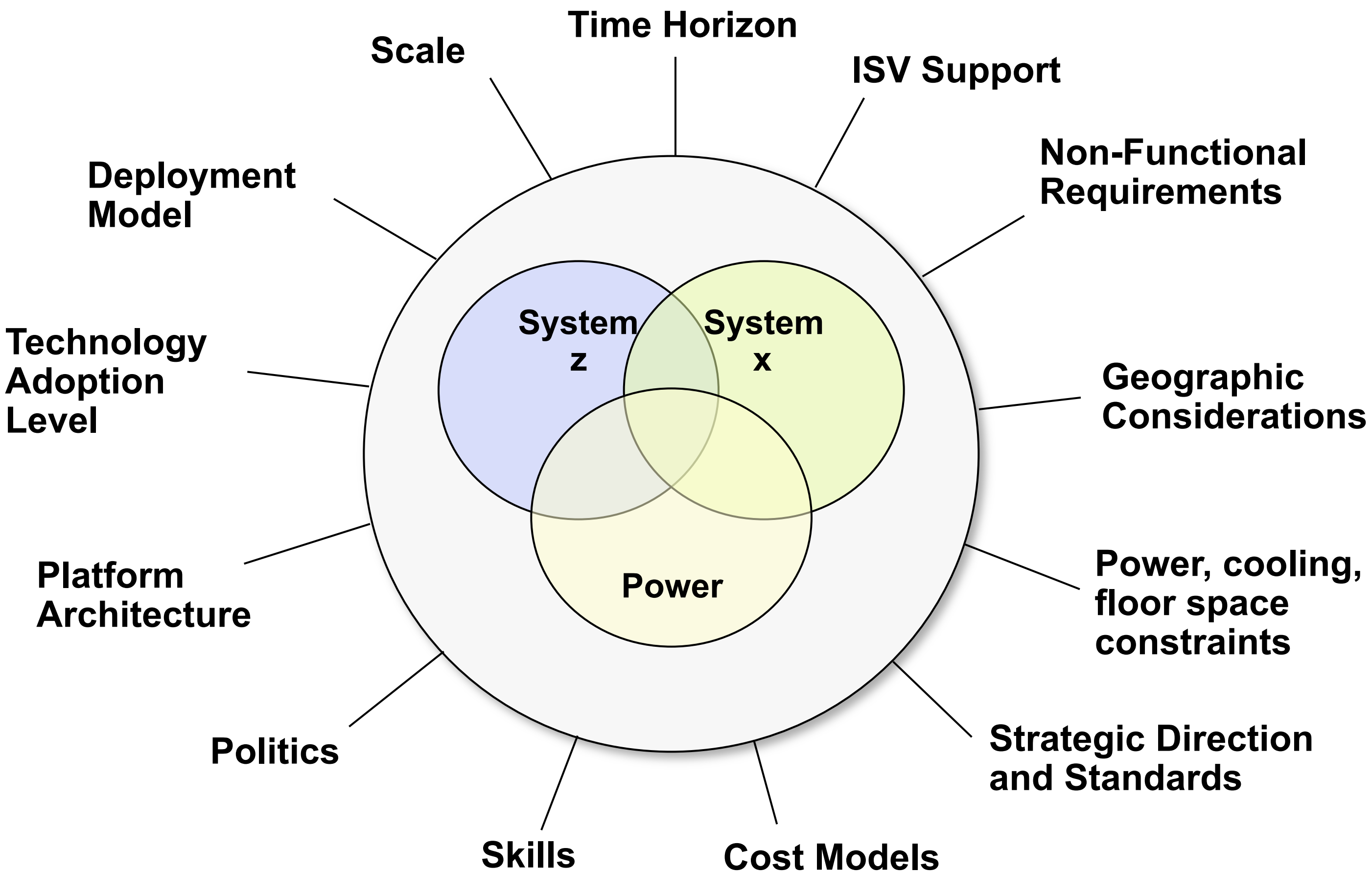
- This presentation covers both platform positioning and selection
  - Positioning: Generalized discussion of platform design & attributes
  - Selection: Evaluating options for workloads in a specific customer context
- By definition there will be exceptions to any generalization
  - Broad technology & market observations – a starting point for discussion
  - Not all customers are the same – primary focus is on medium to large customers
- Agreement on common terminology definitions is key
  - Similar terms can sometimes mean different things for different platforms
  - Speaker will try to define terms – audience should ask if unsure

Many Factors Affect Choice

Would you purchase a family car solely on one factor?

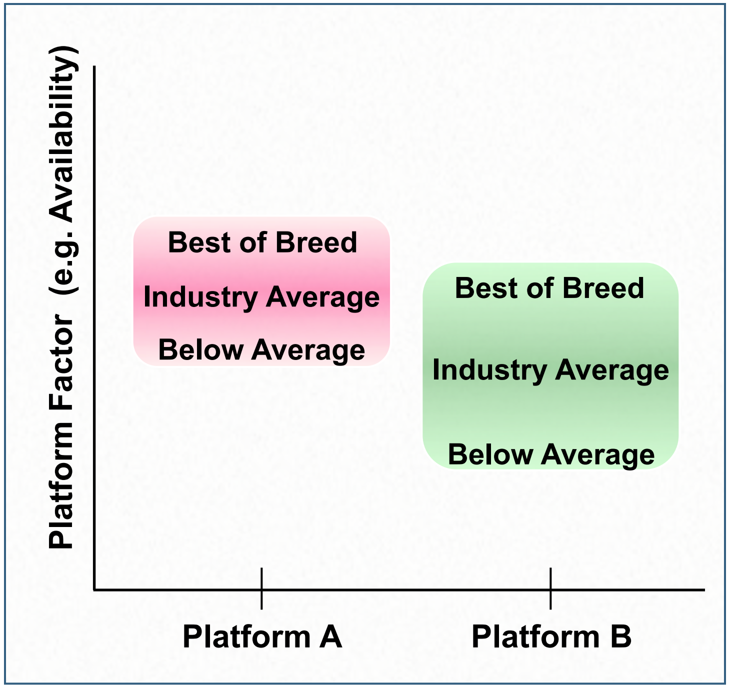
Car	Server Platform
Purchase price	Purchase price
Gas mileage, cost of repairs, insurance cost	Cost of operation, power consumption, floor space
Reliability	Reliability
Safety, maneuverability, visibility, vendor service	Availability, disaster recovery, vendor service
Storage capacity, number of seats, towing capacity	Scalability, throughput
Horsepower	Chip performance
Dash board layout Automatic or manual	Instrumentation and skills
Handling, comfort, features	Manageability
Looks, styling, size	Peer and industry recognition

Selecting a Platform



Local Factors are Important

- Local factors (constraints)
  - Skills
  - Technology adoption levels
  - Platform management practices
  - Number of servers
  - Organizational considerations
  - Data center issues
  - Culture
- Develop comparison metrics
  - Consistent, collectable, usable
- Become best of breed

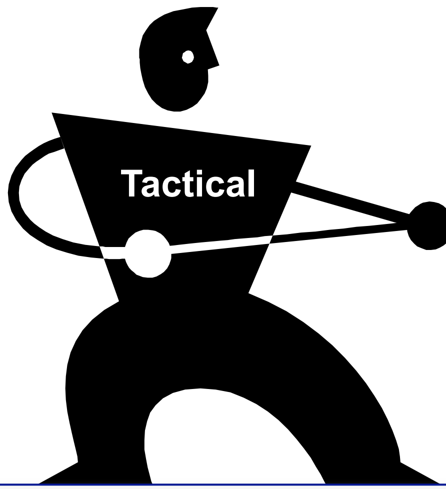


Design Decisions Often Involve Tradeoffs

- Designs involve tradeoffs
  - Cost
  - Availability
  - Throughput
  - Simplicity
  - Flexibility
  - Functionality

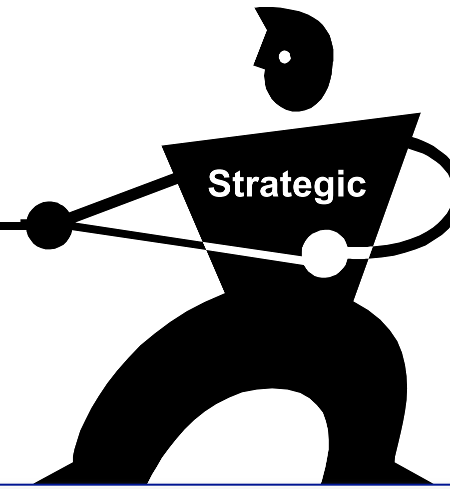
- Designs are different because needs are different

Strategic and Tactical Platform Choices



IT Decision

- Balance tactical & strategic
- Balance of established and visionary patterns
- Mergers and acquisitions can add non-strategic solutions



Tactical Choices

- Can Lower decision costs
- Based on momentum, skills, and legacy
- Narrow focus can lead to sub-optimal solutions

Strategic Choices

- Can lower long term costs
- Can run afoul of legacy
- New technologies may change business process & strategy

Over time all strategic choices become legacy

Reference Architectures

- Pattern for repeated decisions
  - Lower decision making cost
  - Lower implementation variability
- Larger than single decision - unlike a standard
- Based upon
  - Actual implementations
  - Architectural decisions
- Can be long term decision setting

Functional and Non-Functional Requirements

Functional "What it does"	Non-Functional "How well it does it"
• Correct business results • Inputs • Outputs • Behaviors • External interfaces • Screen layouts	• Availability requirements • Transactions per minute • Security requirements • Ease of provisioning and support • Disaster recovery requirements • Future growth

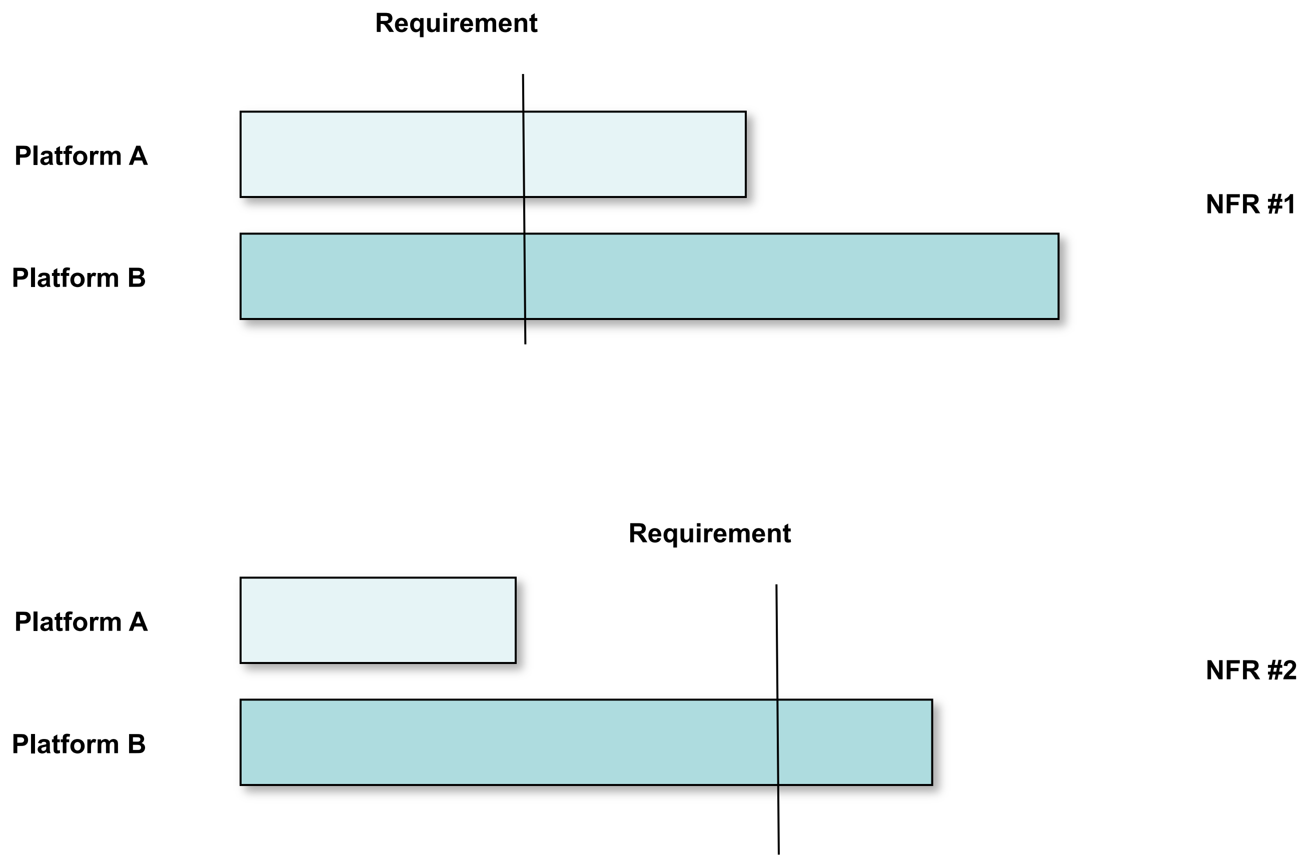
Select platforms based upon non-functional requirements driven by business value

Future Proof

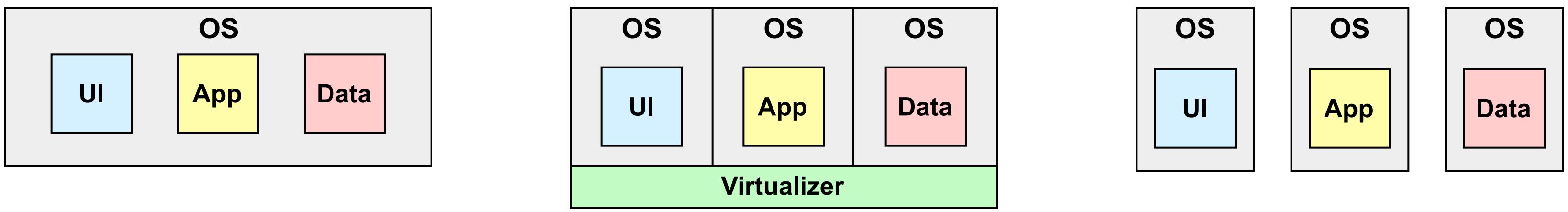
- May select a platform for an unclear future need
  - Potential acquisition or new business volumes
  - More stringent future non-functional requirements
  - Proposed new regulation
  - Money in this year's budget

- Platform features can help mitigate
  - Capacity on demand
  - More scalable servers
  - I/O drawers and/or cages

Select a Platform Based Upon All NFRs



Traditional Deployment Models



Centralized

- Components are all together
- Very granular resource sharing
- OS workload management
- Strongly integrated and stacked

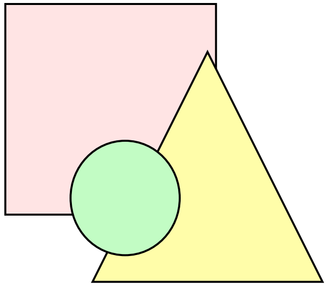
Virtualized

- Components split across virtual images
- Coarser grained resource sharing
- Virtualizer workload management
- Stacked and integrated over network

Dedicated

- Components split across servers
- No resource sharing between servers
- Limited workload management
- Integrated over physical networks

Emerging Deployment Models



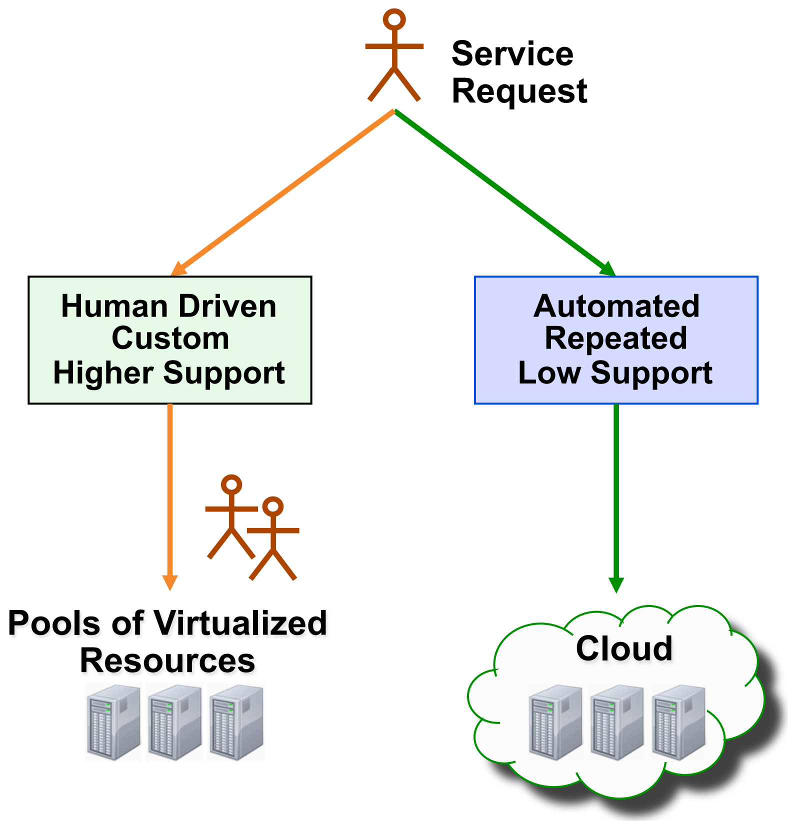
Cloud

- Self service
- Rapid provisioning
- Advanced virtualization
- Flexible pricing
- Elastic scaling
- Standardized offerings
- Network access

Hybrid

- Combination of other deployment models
- Potential uses
  - Intelligent offload
  - Centralized management
  - SOA services
- Transcends computing silos

Clouds vs. Systems Pools



- Automated
  - Capable of automation
  - Limited choice & complexity
  - Small relative to HW footprint
- Repeated
  - Used frequently
  - Low relative effort to automate
- Low on-going support
  - Limited maintenance, capacity planning, performance tuning
  - HW/OS able to dynamically handle needs

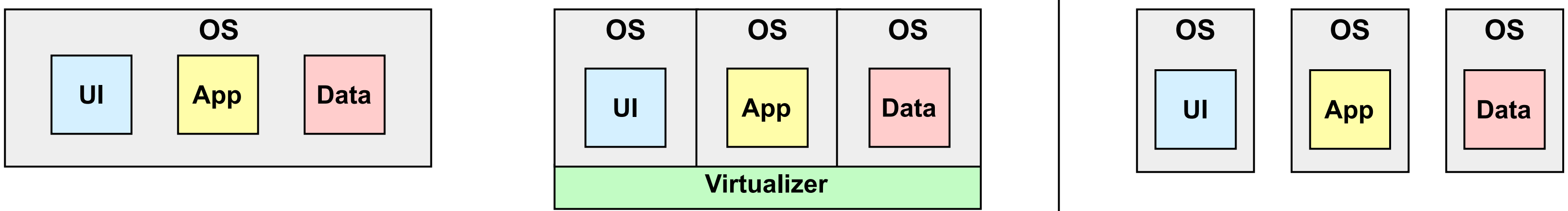
Choosing a Platform – ISV Considerations

- For a given deployment
  - Eliminate platforms that don’t support the ISV
  - Choose a different ISV
  - Get ISV to support platform

- Select platform on non-functional requirements
  - Not just on middleware brand
  - NFRs vary using the same middleware

- Consider platform agnostic middleware
  - NFRs can change over time
  - Agnostic middleware offers more flexibility

Deployment Models and Underlying Hardware



Centralized   Virtualized

- Scalability and performance critical
- Stress on data delivery features
- Large emphasis on single server availability

Dedicated

- Acquisition costs emphasis
- Industry standard components
- Commodity availability features common
- Cost per unit of work is typically higher

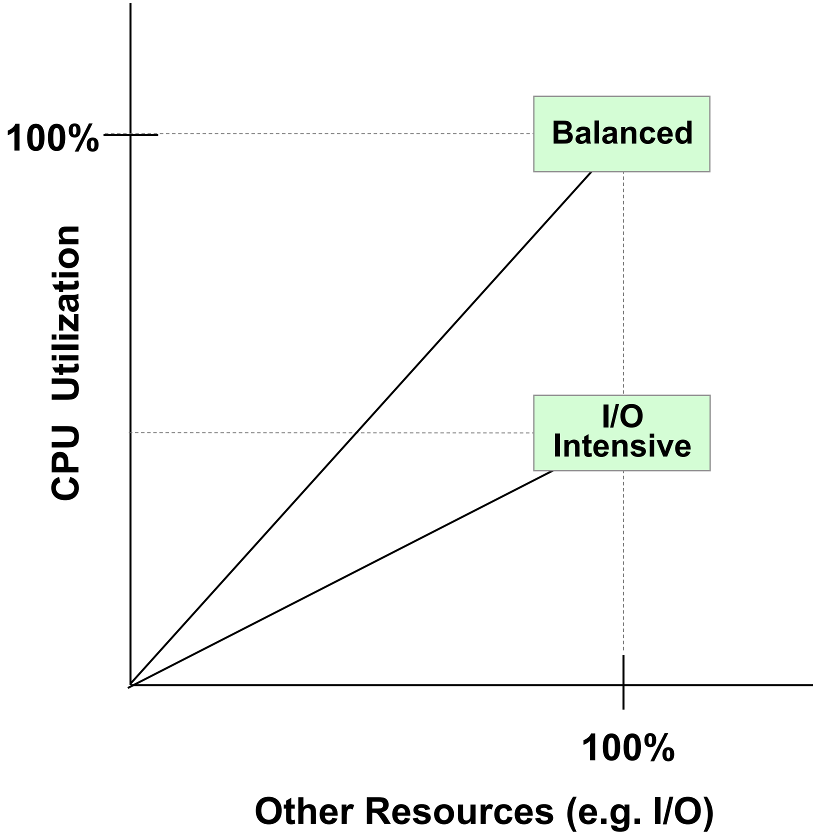
Server Scalability, Utilization, and Throughput

- Throughput measures work
  - Requires performance objective
  - Can be higher with discretionary work
- Factors that affect throughput
  - Cache or data coherence
  - Contention for shared resources
  - Path length and latency
  - Balanced system design
- Mixed workloads can stress platform design
- Isolated capacity can lower throughput
  - Dedicated servers or partitions
  - Passive clusters
  - Separate environments
  - Business decision

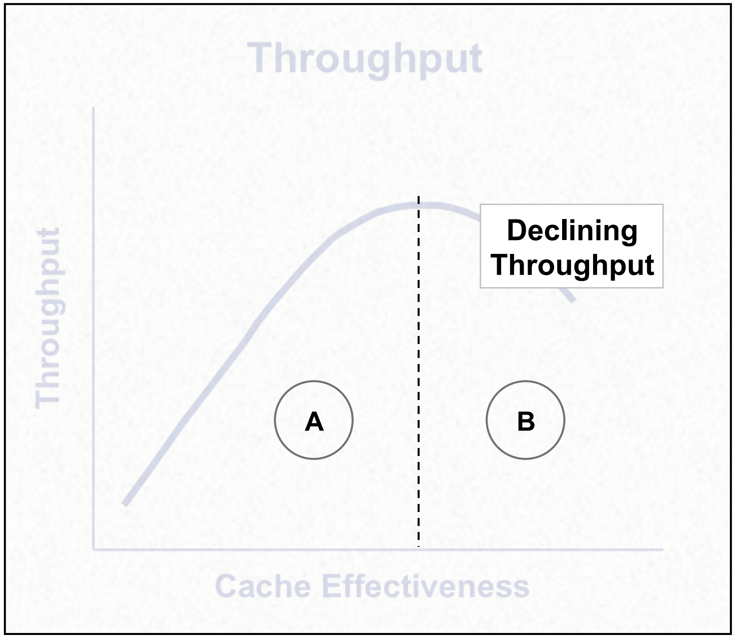
New

Driving High Effective CPU Utilization

- High CPU utilization
  - Minimize core based SW costs
- Must run out of CPU last
- Over size I/O and memory if work is unpredictable



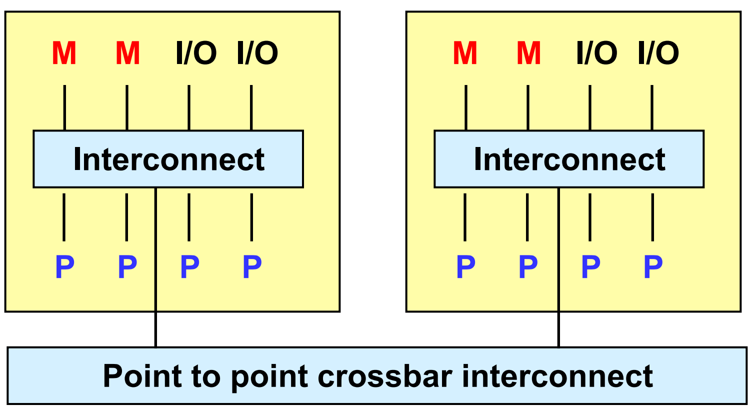
Cache Effectiveness

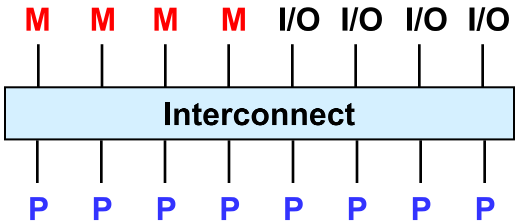


- Throughput can decline when cache is starved
- Area A
  - CPU frequency, memory latency, and threading affect slope
  - Cache is not yet a bottleneck
- Area B
  - Insufficient cache dominates
- Performance affected by
  - Size and distance of cache
  - Working set size and context switch rates

Memory Architectures

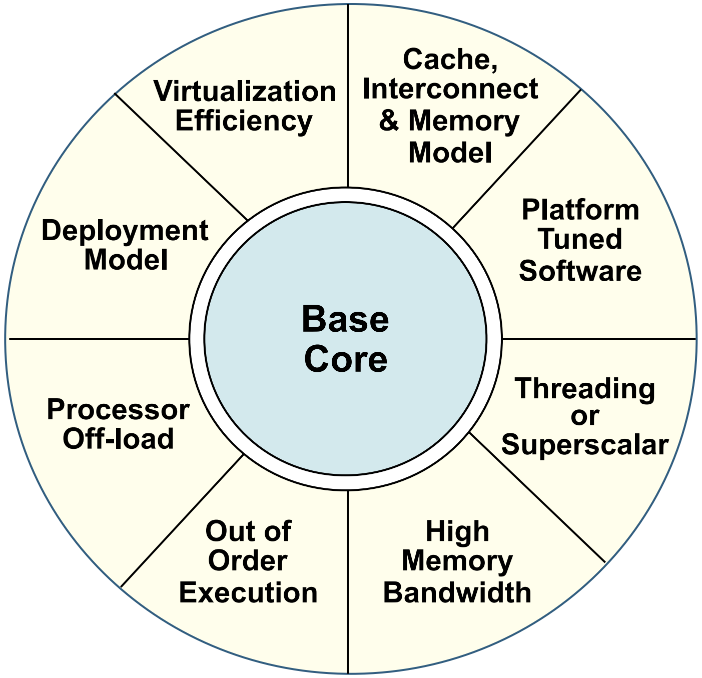
- NUMA Memory Model
  - Non-uniform access to memory
  - Useful for partitioned workloads
  - More components
  - Latency limits throughput
- Flat Memory Model
  - Consistent access to memory
  - Ideal for shared workloads
  - Fewer components
  - Increasingly critical with scale





Relative Server Capacity

- Base core capacity
- Server specific factors
  - Efficiency
  - Additional cycles/capabilities
- Relative server capacity
  - Workloads vary over time
  - Local metrics or relevant benchmarks
- General purpose vs. specialized servers



Workload Attributes and Market Segmentation

Transaction Processing and Database

High Transaction Rates
High Quality of Service
Peak Workloads
Resiliency and Security

Analytics and High Performance

Compute or I/O intensive
High memory bandwidth
Floating point
Scale out capable

Business Applications

Scale
High Quality of Service
Large memory footprint
Responsive infrastructure

Web, Collaboration and Infrastructure

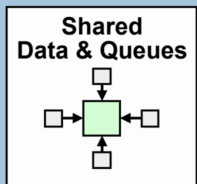
Highly threaded
Throughput-oriented
Scale out capable
Lower Quality of Service

High Level Workload Definition

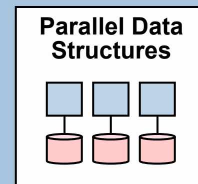
- Workloads are a combination of:
  - Application function: What it does and how it does it
  - Data structure: Data residency, topology, access model
  - Usage pattern: Utilization profile over time, mix of use cases
  - Service level: Non-functional requirements
  - Integration: Interaction between application & data components

- The workload requirements will create varying demands when determining server alternatives

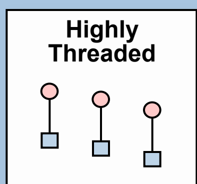
Workload Architectures – More Technical View



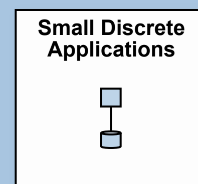
- Shared data and work queues



- Parallel data structures

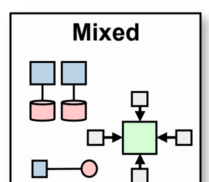


- Highly threaded

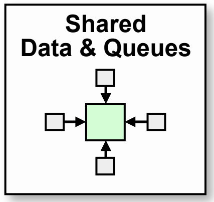


- Small discrete applications

Mixed

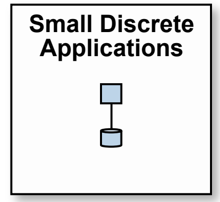


Workload Characteristics



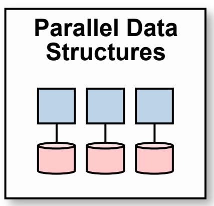
Shared Data & Work Queues

- Single thread performance key
- Benefits from large shared caches
- Fast lock processing
- Hypervisor spin lock detection



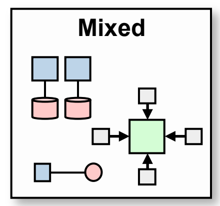
Small Discrete Applications

- Little pressure on any resource
- Minimal memory footprint
- Ripe for virtualization
- May have inactive, low or spiky use



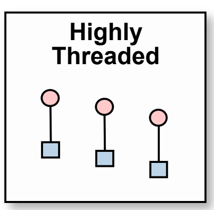
Parallel Data Structures

- High sustained thread use/count
- High memory and I/O bandwidth
- Benefits from large private caches
- Efficient use of dedicated resources



Mixed

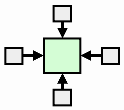
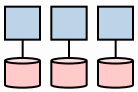
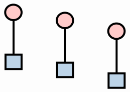
- Different SLAs
- Varying size & number of threads
- Large close caches
- Variable context switch rates

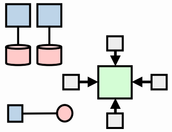


Highly Threaded

- Lots of software threads
- Modest thread interaction
- Benefits from large private caches
- Low latency memory access

Workload Characteristics and Platform Requirements



	Shared Data & Queues	Parallel Data Structures	Highly Threaded	Small Discrete Applications	Mixed
Examples	▪ OLTP databases ▪ N-Tier transaction processing	▪ Structured BI ▪ XML parsing ▪ HPC applications	▪ Web app servers ▪ SAP app servers	▪ HTTP, FTP, DNS ▪ File and print ▪ Small end user apps	▪ z/OS and IBM i ▪ Hypervisors with virtual guests, WPAR
Characteristics	▪ Thread interaction raises contention & coherence delays ▪ Coherency traffic increases memory & cache bus utilization ▪ High context switch rates	▪ Low thread interaction ▪ High memory bandwidth ▪ Low context switch rates	▪ Lots of software threads ▪ Modest thread interaction	▪ Does not pressure any resource ▪ Requires minimal memory footprint ▪ Inefficient on dedicated resources ▪ No shared data	▪ Different SLAs ▪ Varying sizes and number of threads ▪ May be N-Tier or independent ▪ Variable context switch rates
Platform Considerations	▪ Scale on robust SMP ▪ Cluster technology dependent ▪ Large shared caches and wide busses ▪ Fewer, bigger threads	▪ Scale well on clusters ▪ Large private caches ▪ High thread count ▪ High memory and I/O bandwidth ▪ Often on dedicated machines	▪ Scale on large SMP ▪ Can scale on clusters ▪ High thread count ▪ Low latency memory access ▪ Large private caches	▪ Single instances can run on almost any hardware ▪ Small numbers will virtualize on any hardware ▪ Robust SMP allows better virtualization flexibility	▪ Scale on robust SMP ▪ High internal bandwidth ▪ Thread speed and number is workload dependent ▪ Large, close caches ▪ High memory bandwidth

Amdahl's Law

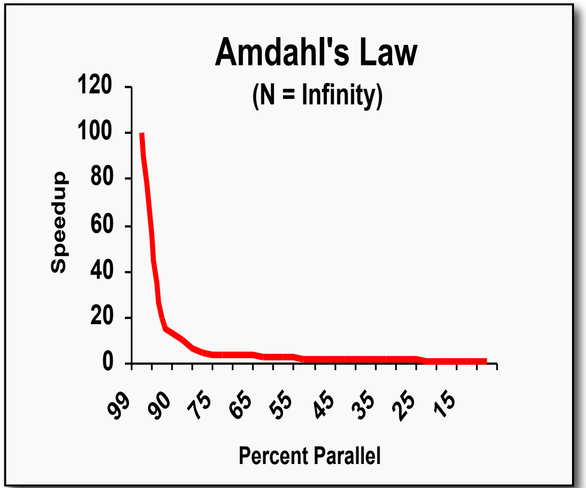
- Limits to parallel applications
  - Not all applications benefit from, or are coded for, more threads
  - Individual thread performance matters
  - Benchmarks often exploit unrealistic parallelism
  - Applies to SMPs and clusters

Amdahl's Law

$$\text{Speedup} = \frac{1}{(1 - P) + \frac{P}{N}}$$

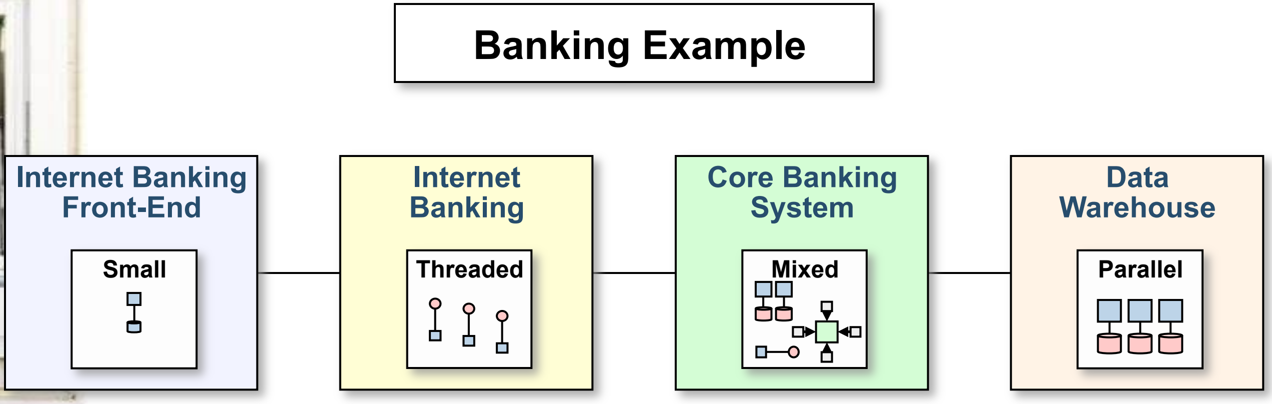
P = Proportion of a program that can be made parallel

N = Number of threads



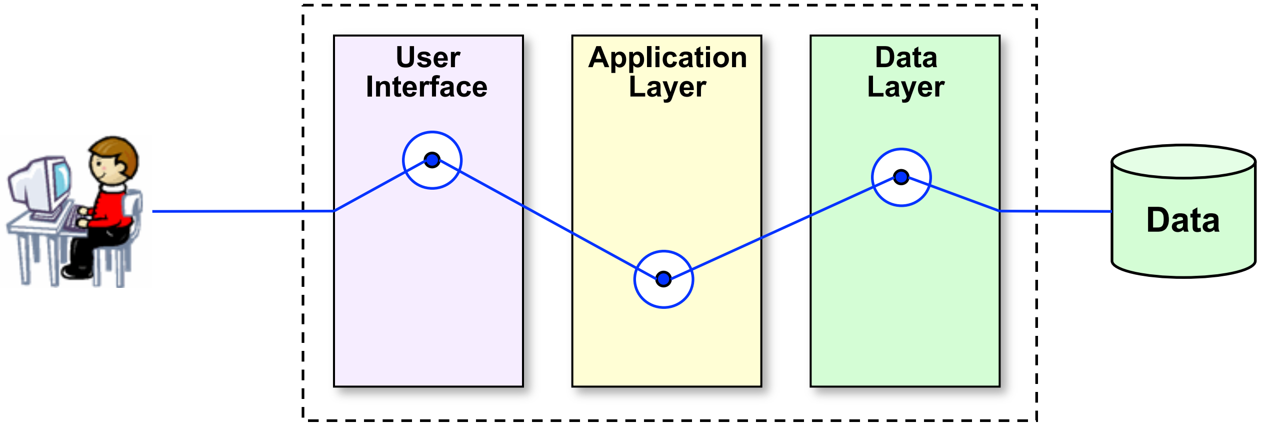
Multiple Platforms May be Appropriate

- A business service
  - May have multiple workload types
  - Can exhibit multiple workload types based on usage patterns
- Impact on selection
  - A mix of optimized platforms may be more cost effective
  - Other local factors and non-functional requirements apply

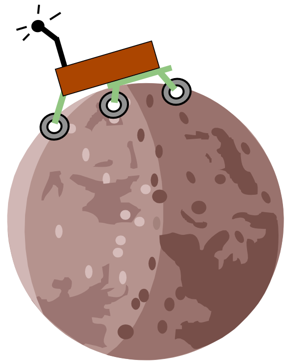


Capacity and Performance - Deployment Models

- Deployment Considerations
  - Network effects
  - Sharing of resources
  - Workload management
  - Multi-programming level
- Impact
  - Total resource requirements
  - Server utilization
  - Performance/capacity mgt
  - Disaster recovery



Automation and Feedback Loops



- **Automation is built on feedback controls**
- **Scaling out lengthens feedback loops**
- **Long feedback loops can result in solutions that are:**
  - Over-provisioned
  - Inconsistent
  - Sluggish

Scope Limitation Leads to Sub-Optimization

- A single application or department view is easiest to understand

- Issues
  - May be driven by politics
  - Runs counter to enterprise IT optimization
  - May make an enterprise view harder to establish
  - Can lead to large hidden costs
  - Server sprawl

- Enterprise wide, scope specific, reference architectures

Commit Scope

All for one and one for all

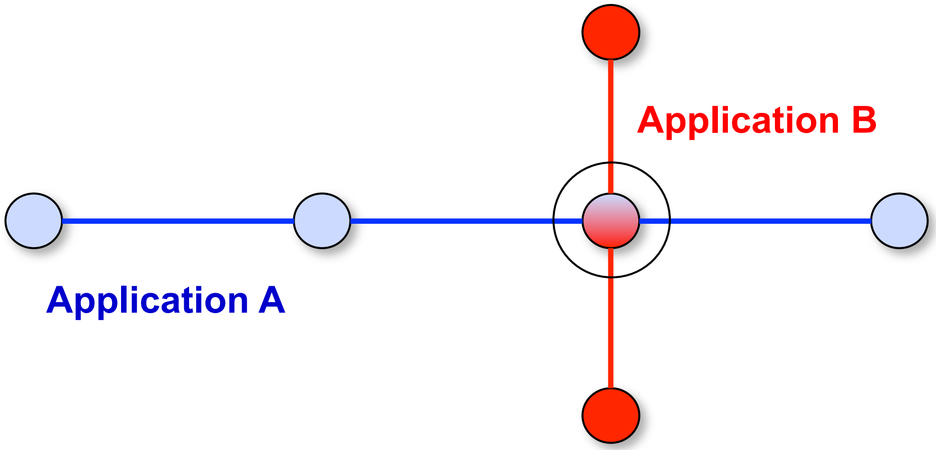
- Commit Scope
  - Data for an application must be in sync and committed together
  - Must also be backed off together
- Impact of Deployment Model
  - Latency increases the time resources are held
  - More parts increases integrity issues during commit scope
- Can be significant in HA and DR scenarios

Interference Analysis

- Interference analysis
  - Few things operate in isolation
- New application or service
  - May impact existing applications
  - May include procedural impacts including life cycle management
  - Can be significant for composite applications
- Non-functional requirements
  - A requirement or change in one NFR can affect another

Non-Functional Inheritance

- Sum of calling applications arrival rates
- Fastest of the calling applications response times
- Highest of calling applications availability
- Non-functional inheritance drives up requirements

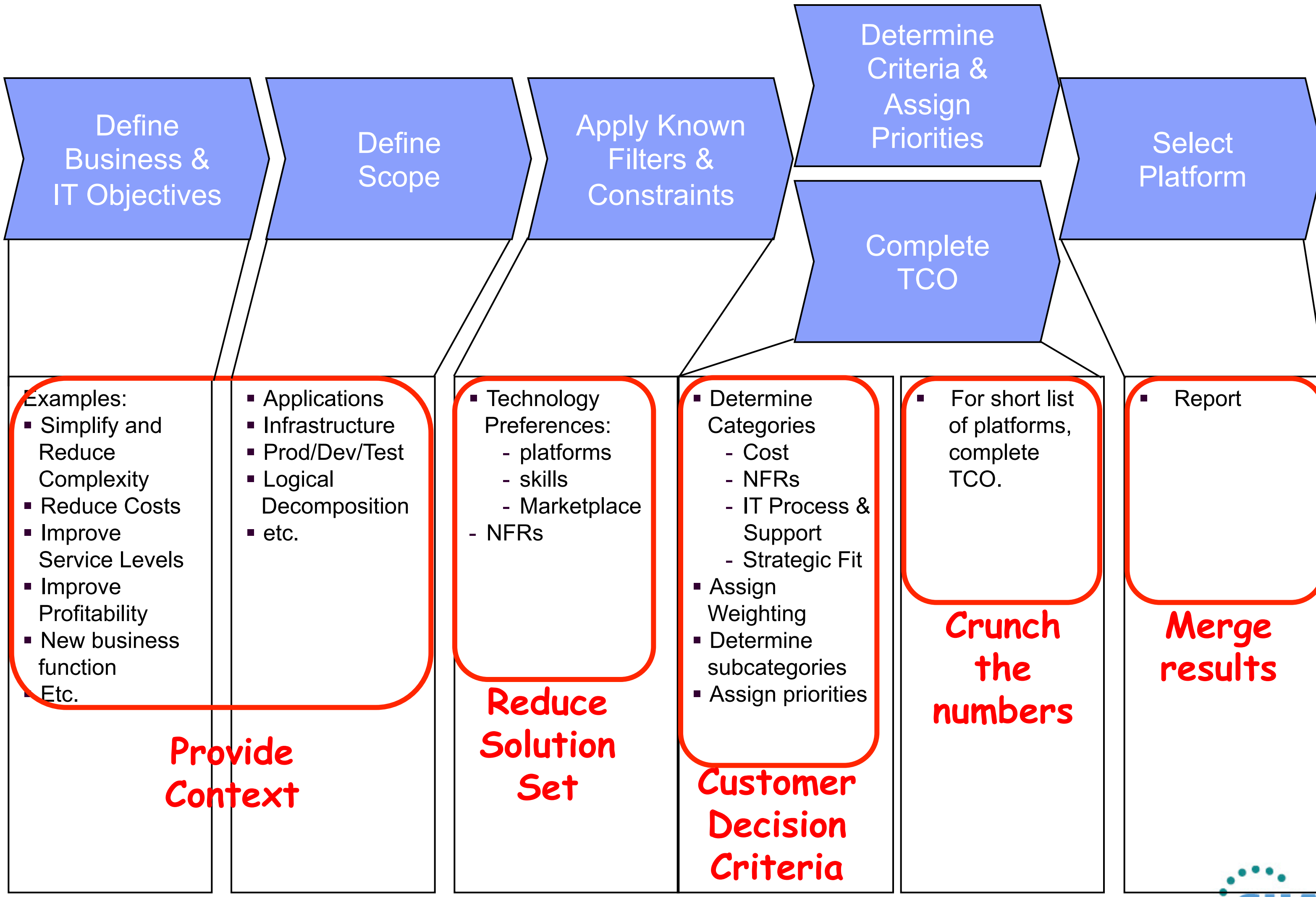


IBM's Consolidation Project

	Linux on System z	AIX on Power	Windows / Linux on System x
ISV Filter	Software available only on a particular platform		
Performance Filters	Low CPU Peak Average memory usage	High CPU Peaks Higher memory usage	
Workload Filters	Transactional I/O Proximity to Data Proximity to Apps	Already virtualized on AIX AIX product development	Already virtualized on Intel Small counts of isolated images Linux not met by System z

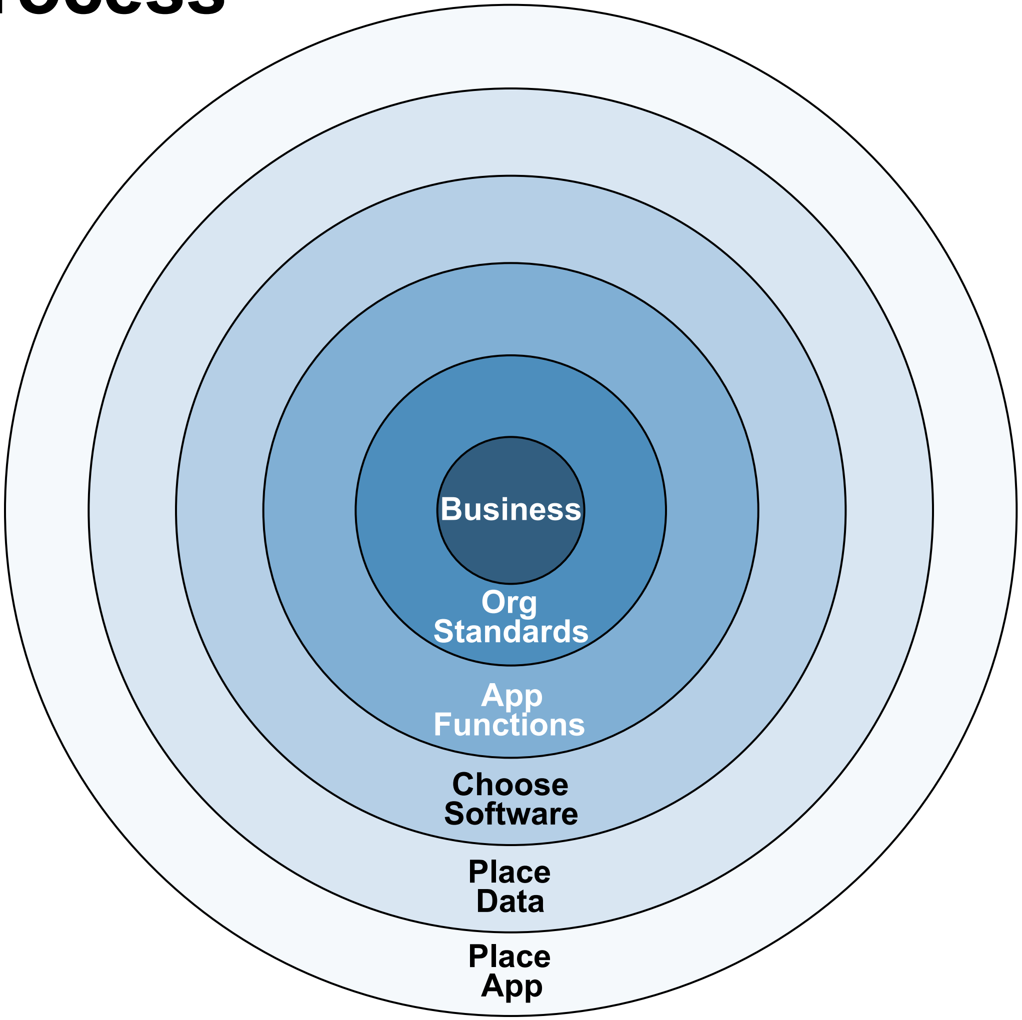
This table reflects an IBM example based upon IBM's local factors

Platform Selection Framework



Platform Selection Process

- Start with business need and scope
- Understand local standards
- Choose software stack
  - Consider operational context
- Select hardware based on NFRs & local factors
  - First data then apps
  - Consider operational context



Example Platform Selection Process

Business & IT Objectives

- Reduced costs
- Reduced complexity
- Improve server levels
- New business function

Define Scope

- Application(s) in question
- Environments considered
- Locations
- Length of decision
- Identify standards, constraints & interference

Build Guidance Weighting Model

For Data then App

- Address local factors
- Evaluate prioritized NFRs
- Assess for interference

Eliminate platforms

Technical Validation (as required)

- Stress test
- Benchmark
- Modeling
- Solution review

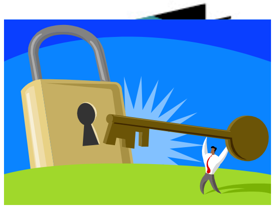
Eliminate platforms

Complete TCO Analysis

Analyze Costs and Weights

Select Platform

Summary of Key Points



- Many factors influence platform selection – a simple matrix does not exist
- Local factors affect platform selection
- Infrastructure size matters
- Each deployment model has its place – virtualize or centralize where possible
- There is no single platform or middleware capacity metric
- Larger servers offer virtualization advantages
- Non-functional requirements are the significant element of platform selection
- Select platforms based upon workload requirements not middleware
- An enterprise wide view provides the best optimization opportunity
- The choice of cost and value elements, along with time horizon, can dictate which platform is considered the lowest cost
- Cost models have different purposes – use the right one for the job