Research Project: BigIron2

Large Scale Coherent Shared Memory Systems for In-Memory Computing

Ike Nassi
Chief Scientist, SAP

08 Feb 2011

Motivation

- We need to innovate without disrupting. (Like your multicore laptops.)
- Large memory technology is poised to take off, and it needs appropriate hardware.
- High Performance Computing (HPC) has yielded results that can now be applied to Enterprise Software.
- We have a lot of cores now and better interconnects.
- We can “flatten” the layers and simplify.
- We can build systems that improve our software *now* without making *any* modifications.
- Nail Soup argument: But, if we are willing to modify our software, we can win bigger. But we do this on our own schedule.

SAP and Real Real-Time Computing

Real real-time computing is possible
because of in-memory computing

In-Memory Computing

Technology that allows the processing of **massive quantities of real time data** in main memory to provide real time decision making and analytics.

SAP Products With In-Memory Computing

Introducing SAP High Performance Analytic Appliance (HANA)

Real Real-time

- Sub-second update latency
- No materialized views

Fast

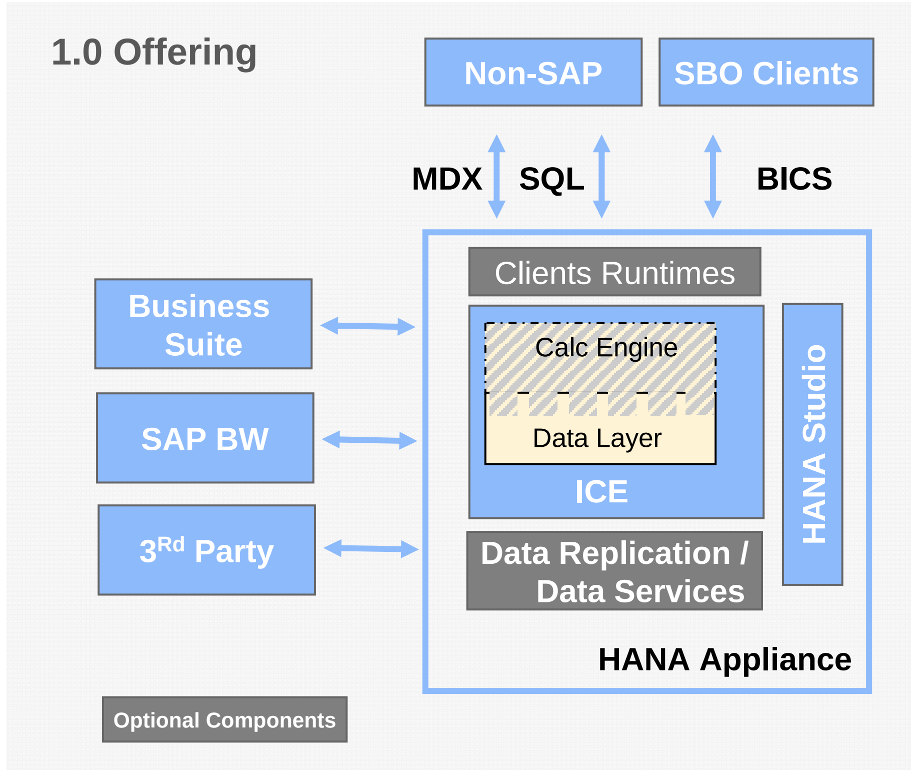
- Native multi-core & MMP support
- Full featured in-proc calc engine
- “BWA on steroids”

Simple & Easy

- Pre-configured appliance
- Modeling based on SBO Information Designer (“universe”)
- Packaged SAP content

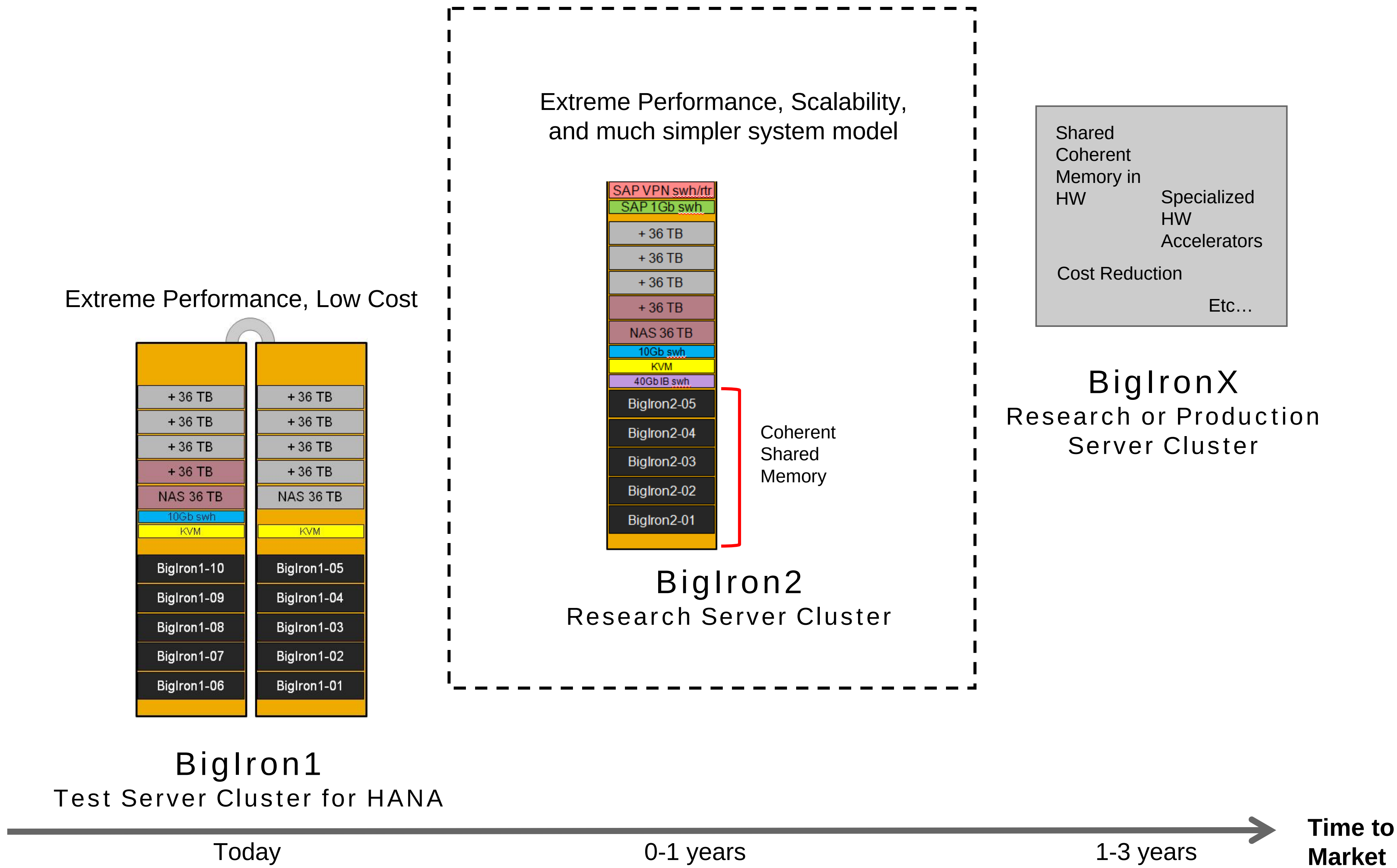
Open

- Full ANSI 92 SQL
- MDX



External

BigIron2 is the second system on a path toward cost-effective, high performance in-memory computing

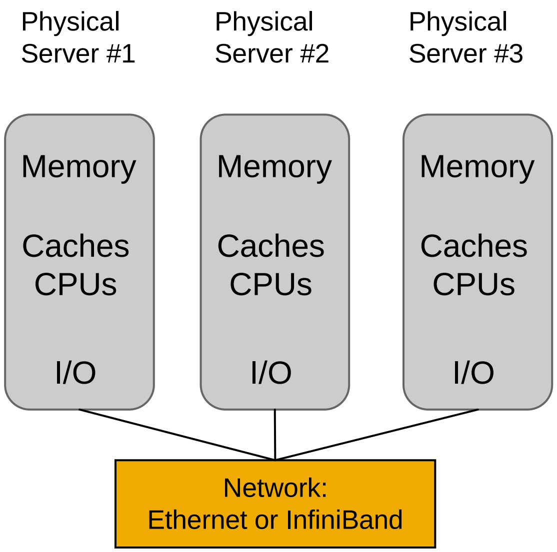


External

Coherent shared memory (CSM)

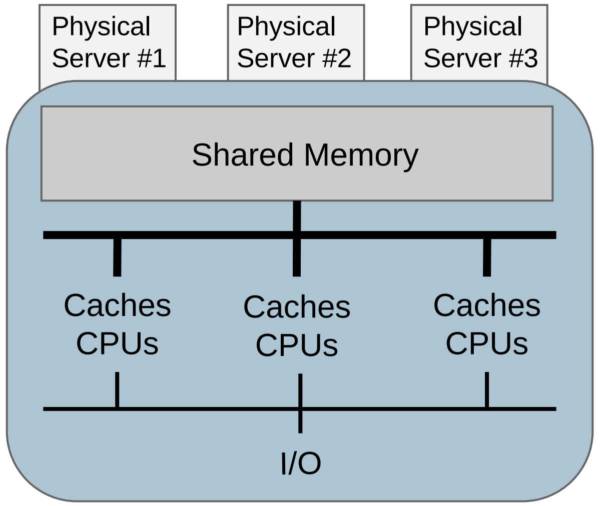
Provides the ability to build a scalable SMP (Symmetric Multi-Processor) system with a uniform and coherent memory addressing architecture that can scale to 10's of terabytes of directly accessible, random access, primary memory. CSM is also called Cache Coherent Non-Uniform Memory Access (ccNUMA).

Traditional server clusters – Distributed memory



- Processes are loosely coupled through a physical network
- Application that needs to utilize more processing, memory, or I/O than those present in each server must be programmed to do so from the beginning

Server Cluster with Coherent Shared Memory



- Server clusters can be physically and logically treated as one "large" server via hardware and/or software solution
- Application can use any resource (processors, memory, I/O) in the system as a virtualized resource

Source: Numascale

Taking In-Memory Computing Seriously

- Basic Assumptions and observations
  - Hard disks are for archives only. All active data must be in DRAM memory.
  - Data locality is essential. Otherwise CPUs are stalled due to too many cache misses
  - There are many levels of caches
- Problems and Opportunities for In-Memory Computing
  - Addressable DRAM per box is limited due to processor physical address pins.
    - ***But we need to scale memory independently from physical boxes***
  - Scaling Architecture
    - Arbitrary scaling of the amount of data stored in DRAM
    - Arbitrary and independent scaling of the number of active users and associated computing load
  - DRAM access times have been the limiting factor for remote communications
    - Adjust the architecture to DRAM latencies (<100 ns?)
    - InterProcess Communication is slow and hard to program (latencies are in the area of 0.5-1ms)

Coherent Shared Memory – The Alternative to Remote Communication

- Uses high-speed, low latency networks (Optical copper/fiber with 40Gb/s or above)
  - Typical latencies of this are in the area of 1-5 µsec
  - Throughput is higher than the CPU can consume
  - L4 cache needed to balance the longer latency on non-local access (cache-coherent non-uniform memory access over different physical machines)

- Separate the data transport and cache layers into a separate tier below the operating system- *never seen by the application or the operating system!*

- Applications and database code can just reference data
  - The data is just “there”, i.e. it’s a load/store architecture, not network datagrams
  - Application level caches are possibly not necessary – the system does this for you.
  - Streaming of query results is simplified, L4 cache schedules the read operations for you.
  - Communication is much lighter weight. Data is accessed directly and thread calls are simple and fast (higher quality by less code)
  - Application designers do not confront communications protocol design issues
  - Parallelization of analytics and combining simulation with data are far simpler, enabling powerful new business capabilities of mixed analytics and decision support at scale

Implications of Coherent Shared Memory

- On Application Servers
  - Access to database can be “by reference”
  - No caches on application server side. Application can refer to database query results including metadata, master data etc. within the database process.
  - Caches are handled by “hardware” and are guaranteed to be coherent.
  - Lean and fast application server for in-memory computing
- On Database Code
  - A physically distributed database can have consistent DRAM-like latencies
  - Database programmers can focus on database problems
  - Data replication and sharding are handled by touching the data and L4 cache does the automatic distribution
- In fact, do we need to separate application servers from database servers at all?
- No lock-in to fixed machine configurations or clock rates
- No need to make app-level design tradeoffs between communications and memory access

But what about…

- Distributed Transactions?
  - We don't need no stinkin' distributed transactions!
- What about traditional relational databases?
  - In the future, databases become data structures!
- Well, not really. Just wanted to make the point. (grant me some poetic license here)
- Is it Virtualization?
  - In traditional virtualization, you take multiple virtual machines and multiplex them onto the same physical hardware. We're taking physical hardware instances and running them on a single virtual instance.
- Why not build a mainframe?
  - It **is** a mainframe

The business benefits of coherent shared memory

Improved in-memory computing performance at dramatically lower cost

- The ability to build high performance “mainframe-like” computing systems with commodity cluster server components
- Ability to scale memory capacity in a more in-memory computing-friendly fashion
- Simplified software system landscape using system architecture that can be made invisible to application software

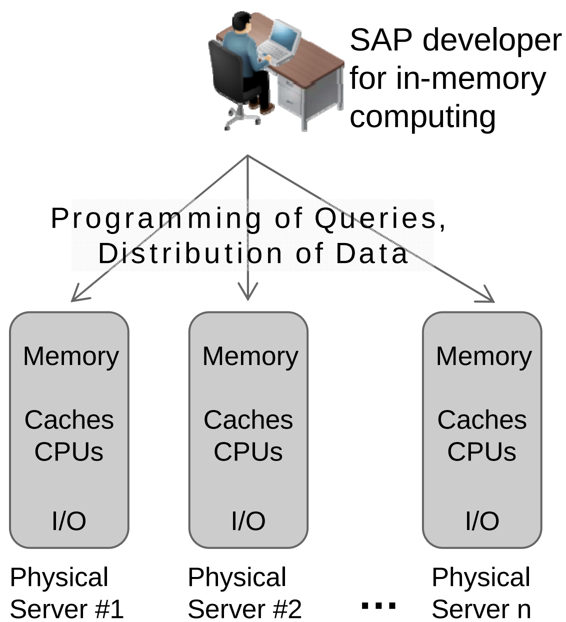
Minimize changes to SAP applications

- Enables SAP applications to scale seamlessly without changes to the application code or additional programming effort
- With coherent shared memory, the bulk of SAP’s developers can develop as they do today and let the underlying hardware and lower level software handle some of the resource allocation, unlike today.

A simpler programming model

Before

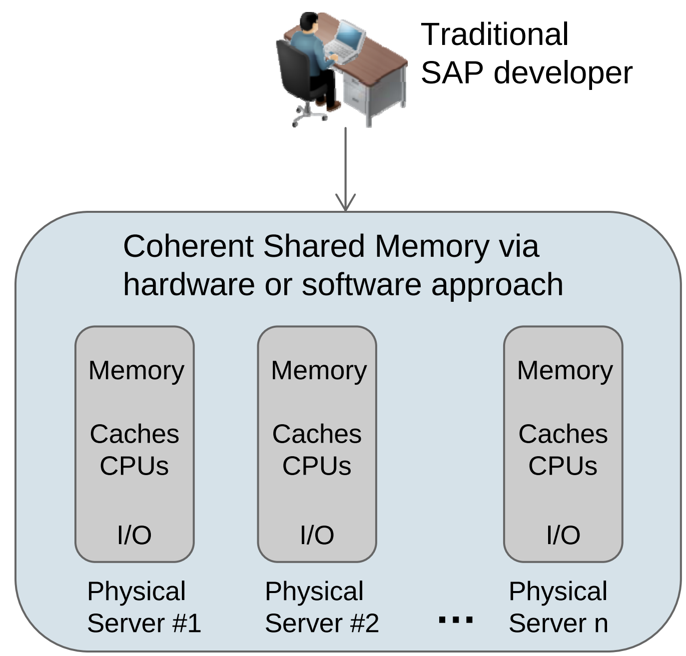
Traditional server clusters – Distributed memory



- Developers need to distribute queries and data across physical servers; a difficult design tradeoff *because access to data requires mastering complex communications protocols*
- Design trapped at a single “scale” – platform growth forces redesign every couple years
- Sufficiently skilled programmers will be scarce

After

Server Cluster with Coherent Shared Memory



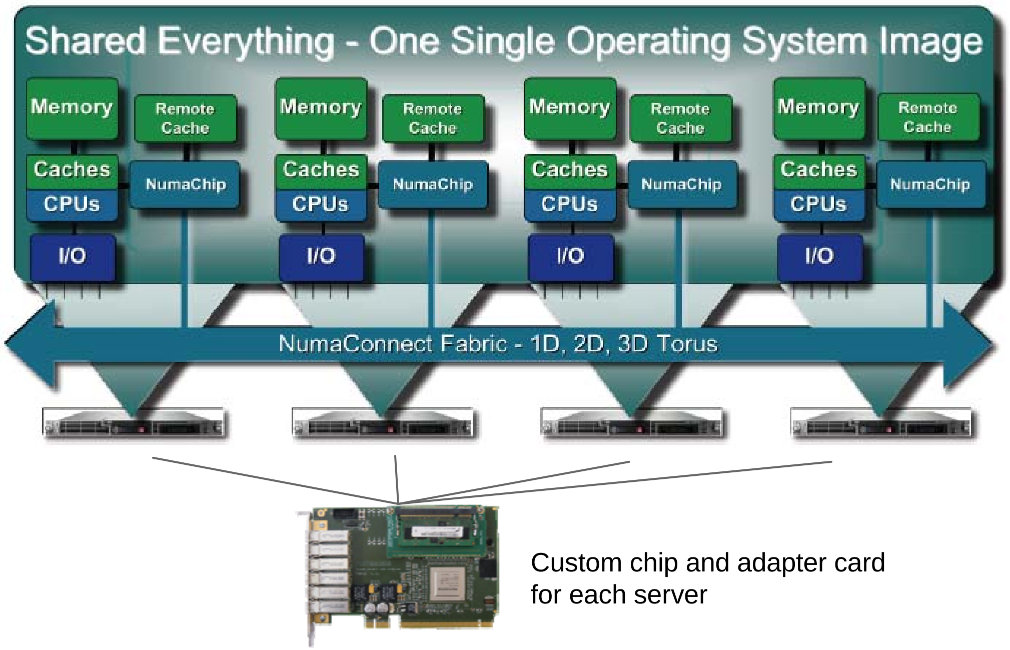
- Developers can treat system as one “big” server and let the operating system and lower level hardware/software handle the problem
- Initial design is timeless – hardware scaling handled below app design layer
- Developers do not need additional skills for in-memory computing

External

Hardware and software approaches to coherent shared memory

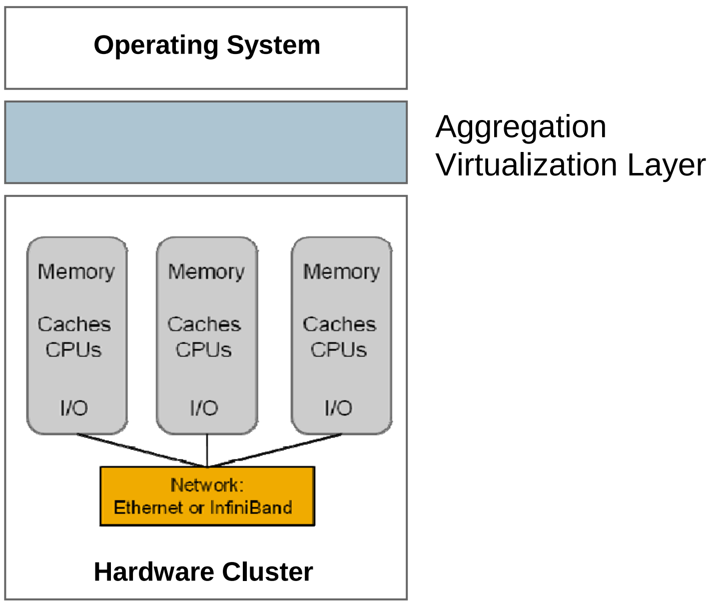
Hardware Approaches

Numascale Example



- Custom and proprietary chipsets (e.g. NumaChip in diagram), with software and commodity interconnects such as InfiniBand, aggregate compute, memory and I/O capabilities of each system

Software Approaches



- Hardware approach is replicated in software
- Software aggregates the compute, memory, and I/O capabilities of each system and presents a unified virtual system to both the OS and the applications running above the OS via a software interception engine

Multiple companies are developing and/or delivering relevant solutions

Hardware and Software Solutions

Huawei: Via 3 Leaf Systems acquisition – HyperTransport approach

ScaleMP: Start-up with software-based approach - vSMP

numascale: Start-up with SMP adapter card

HP: Hardware-based node-controller solution

SGI: Hardware-based node-controller solution

Processor Companies

AMD / HyperTransport Consortium: HyperTransport extensions

Intel: QPI (Quick Path Interconnect) Technology extensions

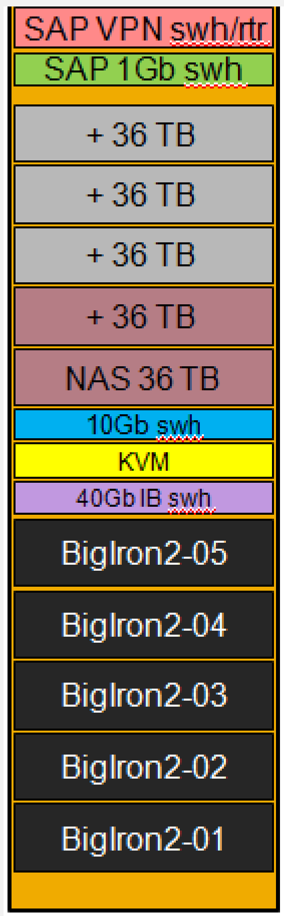
BigIron 2: The system we have architected and built via leading-edge, standard, cluster server components

System Specifications

- 5 x 4U Servers (4 Intel XEON x7560 2.26Ghz)
  - **160 cores** (32 Cores/Server)
  - **5TB memory** (64 x 16MB DDR3/Server)
  - 30TB SSD (solid state disk) storage
- 5 Networks
  - VPN of ScaleMP (40-160GbIB)
  - VPN of Server Cluster (10GbE)
  - VPN of Storage Array (10GbE)
  - VPN of SAP Internal Network (10MbE metered)
  - Firewalled GW to Internet (1GbE Expandable)
- 1 NAS (72TB Expandable to 180)
- 1 x 48U Rack
- System Software
  - SLES11 Linux OS Licenses
  - ScaleMP vSMP Licenses
- Lower System cost

Big Iron 2

Extreme Performance, Scalability, and much simpler system model
Research Server Cluster



- Large shared coherent memory (5TB) across servers via Scale MP
- 160 cores(320 HT)

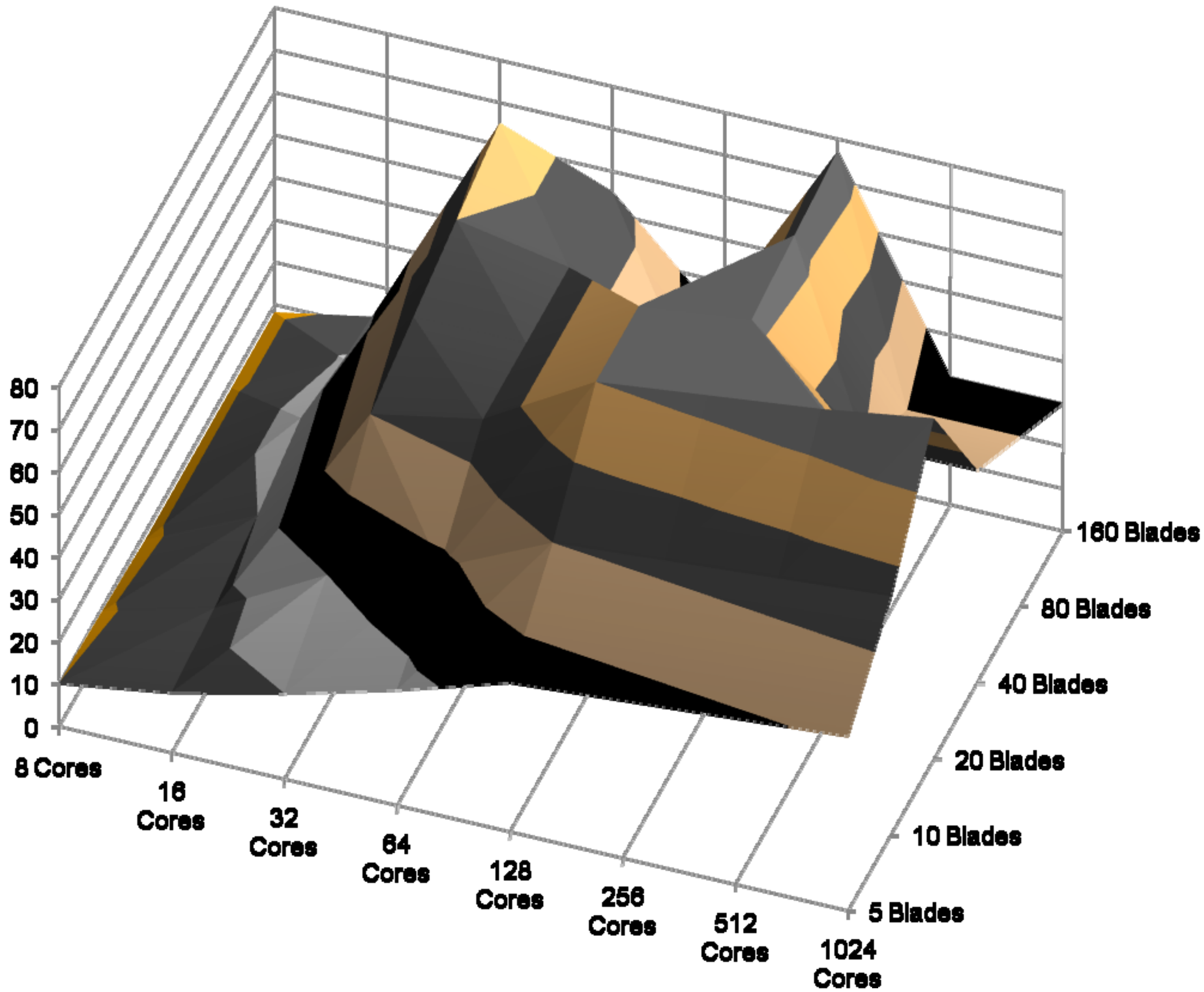
Architecture, Assembly, & Hosting

System architecture:	SAP Technology Infrastructure Research Practice
Assembly and Test:	Colfax International
Hosting:	Bay Area Internet Solutions, Santa Clara, CA

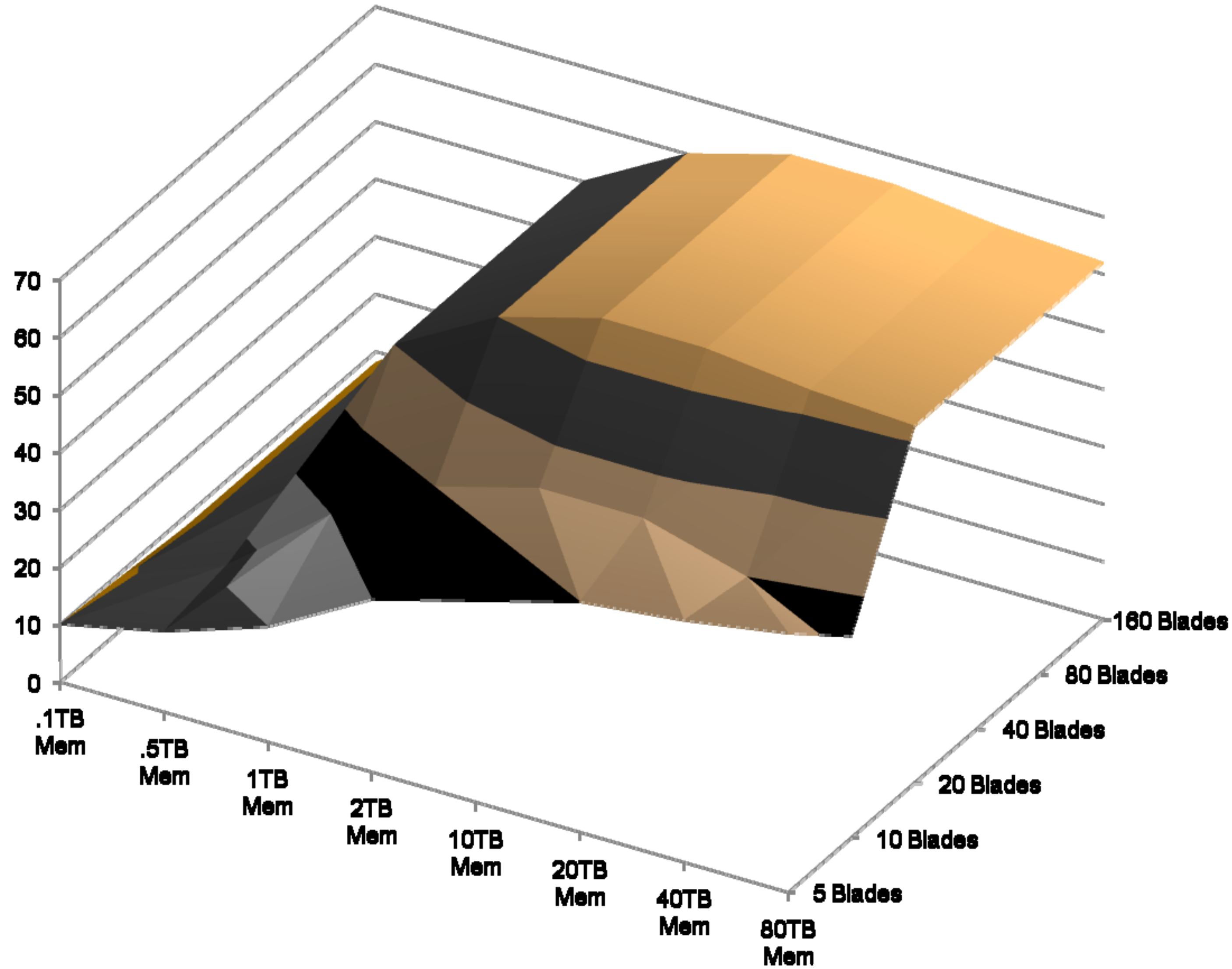
Open Research Questions

- What is the most applicable and realistic approach for SAP? (e.g. in hardware vs. in software)
- Is a software approach even feasible given long-range hardware capabilities and performance estimates?
- What is the right size of L4 cache? What are the working sets? Managing all cache levels.
- What are the interconnect options and latency characteristics. Tradeoffs?
- Are fewer faster sockets/board better than more sockets?
- What are the operational issues, including DB load, errors, failover, resiliency, scale, etc.

Sample BI2 Performance Matrix

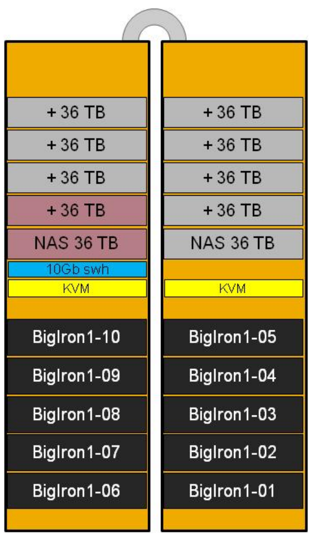


Sample BI2 Performance Matrix



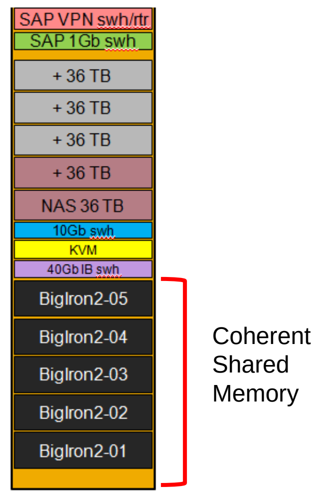
Can we scale further?

Extreme Performance, Low Cost



BigIron1
Test Server Cluster for HANA

Extreme Performance, Scalability, and much simpler system model



BigIron2
Research Server Cluster

Shared Coherent Memory in HW
Specialized HW Accelerators
Cost Reduction
Etc…

BigIronX
Research or Production Server Cluster

Today — 0-1 years — 1-3 years → **Time to Market**

Questions?