www.qct.io

5G AI

Optimizing HPC System Architecture through Workload-Driven Design Strategies

Stephen Chang

Agenda

- Introducing the significance of workload-driven design in maximizing HPC system efficiency
- Key architectural considerations for scalability and performance in optimizing HPC system architecture
- Reference architecture and best practices in workload-driven infrastructure for HPC/AI workloads

Infrastructure for HPC / AI / DA Workloads

Manufacturing

Automotive

Healthcare

Financial

Higher Education & Research

Numeric Weather Prediction | Computation Fluid Dynamics | Molecular Dynamics | Quantum Chemistry | Electronic Structure | Next Generation Sequencing | Finite Element Analysis | Dynamic Structure Analysis

AI | BI | AR/VR | EDA | SCADA | OLTP / OLAP | CAD / CAM / CAE | Data Lake / DW | HFT / RM

Industrial Workloads

Traditional Monolithic Architecture

Bare-metal

HPC workloads | *AI workloads* | *DA workloads*

Pain Points

- Limited human resources to manage multiple system
- Inefficiency for the resource allocation
- Cost more and without synergy

Modern Heterogeneous Architecture

Bare-metal | Virtual Machine | Container

Converged HPC & AI & DA Workloads

Benefits

- Easy Management
- Effectively handle evolving workloads
- Cost-effectiveness and reduce TCO

What is Workload-Driven Design?

Significance of workload-driven design in maximizing HPC system efficiency

- Workload-Centric Approach
- Performance Optimization
- Resource Allocation
- Customization

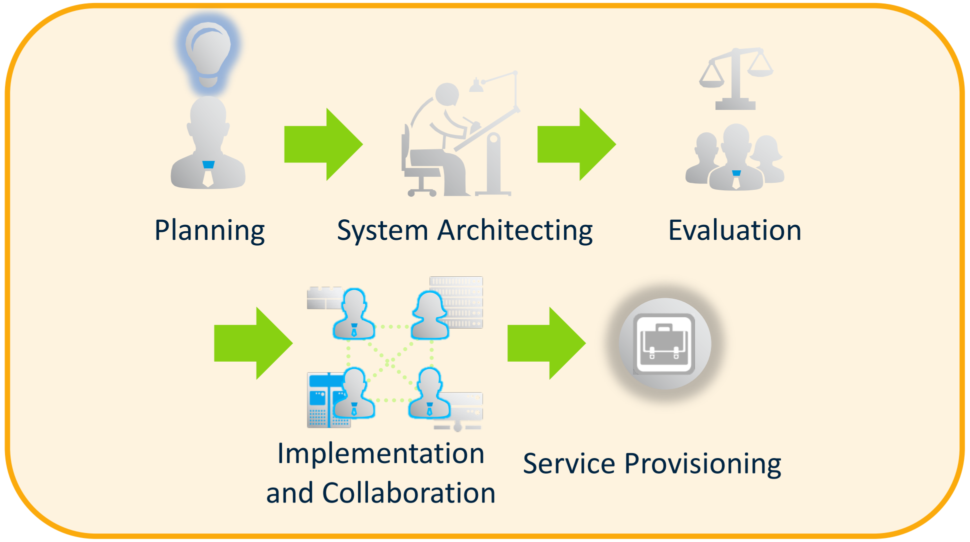
Designing HPC/AI systems based on workloads

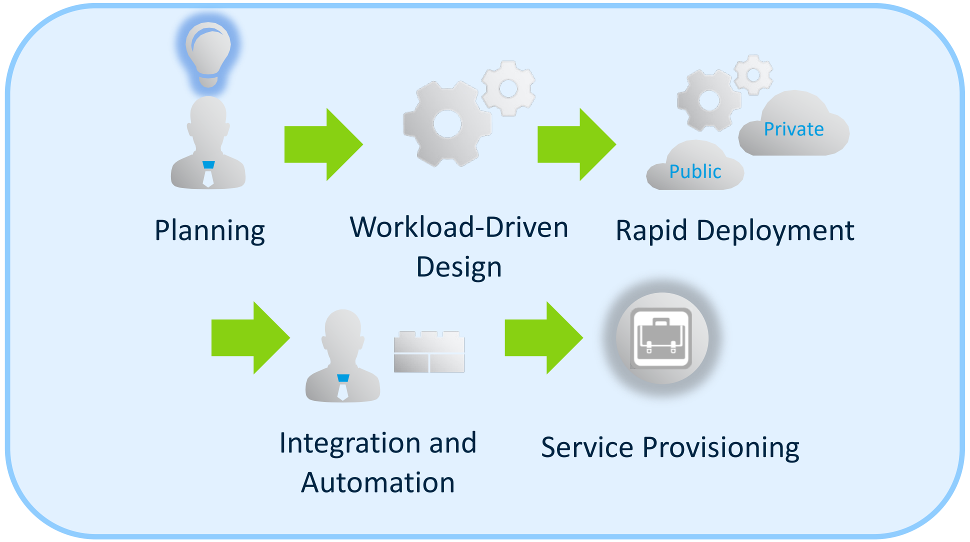
- Scalability
- Performance Bottlenecks
- Heterogeneous Architectures
- Load Balancing
- Adaptive Strategies

Workload-driven design Improves performance, scalability, and resource utilization

- Improved Performance
- Efficient Resource Utilization
- Cost-Effectiveness
- Future-Proofing

System-Centric Design vs Workload-Driven Design



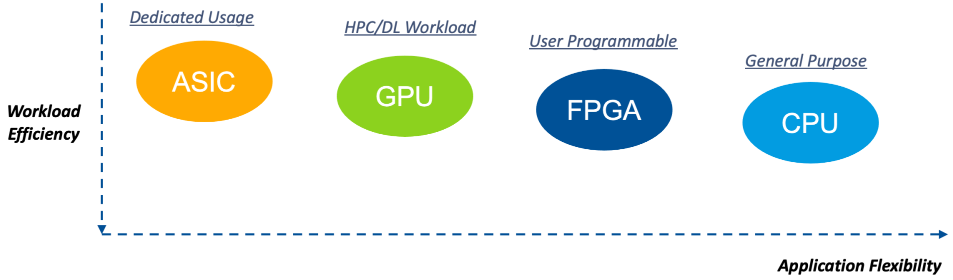


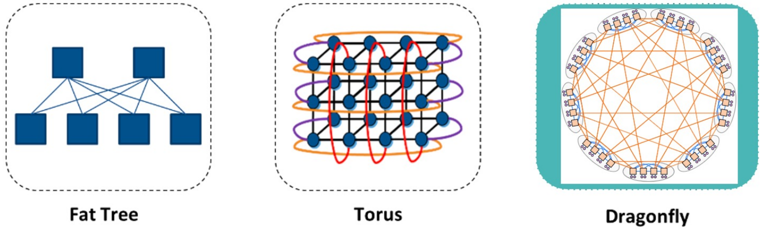
	Traditional System-Centric Design	Modern Workload-Driven Design
Pros	• **Simplicity**: *Easy to deploy and manage* • **Familiarity**: *Established design principles and practices* • **Cost-effectiveness**: *Lower upfront costs*	• **Optimized Performance**: *Tailored architecture for specific workloads* • **Scalability**: *Efficiently scales to handle increased workload demands* • **Efficient Resource Utilization**: *Allocates resources effectively*
Cons	• **Suboptimal Performance**: *Not fully leverages workload characteristics* • **Limited Scalability**: *May have performance limitations and resource constraints* • **Inefficient Resource Utilization**: *Resources may not be optimized for specific workloads*	• **Complexity**: *Requires specialized knowledge and expertise* • **Customization Effort**: *Requires additional time and effort* • **Potential Higher Costs**: *Investments in specialized hardware and software*

Key Architectural Considerations

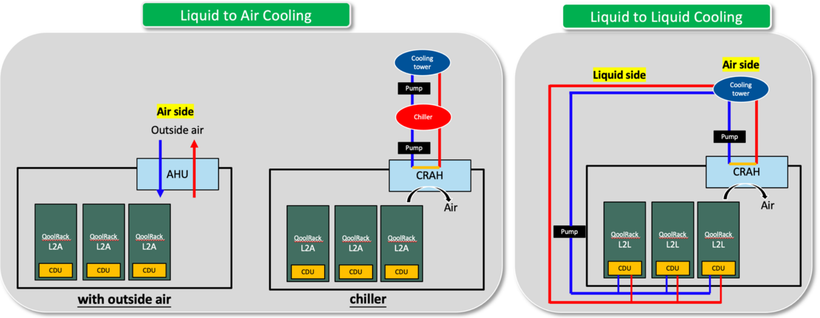
- Understand the characteristics and requirements of different workloads to tailor the system architecture
  - **Workload Types**
    - Identify different types of workloads such as scientific simulations, data analytics, machine learning, and graphics rendering.
    - Each workload type has unique computational patterns and requirements.
  - **Computation Characteristics**
    - Study the computational requirements, including the nature of computations (e.g., floating-point intensive, integer operations), parallelism potential, and load balance.
  - **Memory Requirements**
    - Determine the memory demands of the workload, including the working set size, memory access patterns, and data locality
  - **Communication Patterns**
    - Analyze the communication patterns between compute nodes or processes within the workload.
    - This includes examining the volume, frequency, and patterns of data exchanges.

- Highlighted Considerations:
  - **Processor architecture** and selection (e.g., multi-core, many-core, accelerators)
  - **Memory hierarchy and bandwidth** requirements
  - **Interconnect technologies and topologies**
  - **Storage system design** (e.g., local, distributed, parallel file systems)
  - **Power and cooling** considerations for high-density and high-performance systems





Storge Type	File	Block	Object
Cost	++	+	+++
Performance	++	+++	+
Scalability	++	++	+++

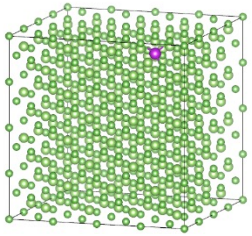


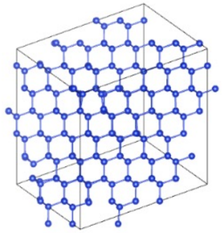
Architectural Consideration for CPU

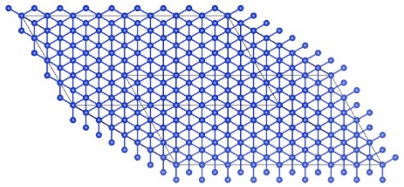
VASP

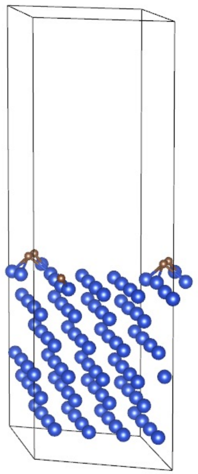
The Vienna Ab initio Simulation Package (VASP) is a computer program for atomic-scale materials modeling based on first principles.

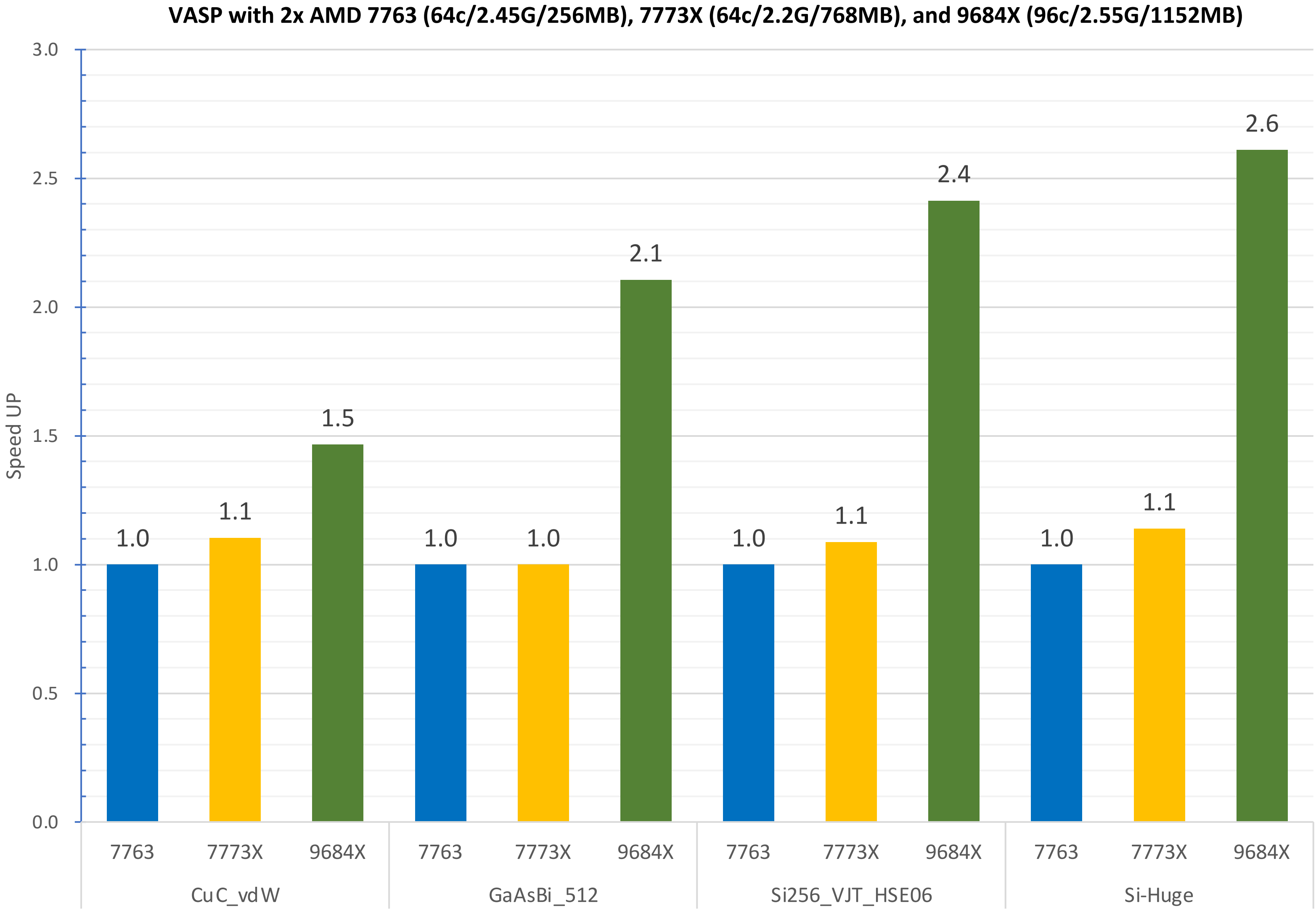
- Version: 6.3.2
- Release Date: 28th June 2022
- Web site : https://www.vasp.at/
- Compile with : Intel oneAPI 2023.1.0
- Test case : CuC_vdw, GaAsBi-512, Si256-VJT-HSE06 and Si-Huge
- VASP work support : Jyh-Pin Chou, Associate Professor, Dep. of Physics, NCUE


GaAsBi_512


Si256_VJT_HSE06


Si-Huge


CuC_vdw



Architectural Consideration for GPU

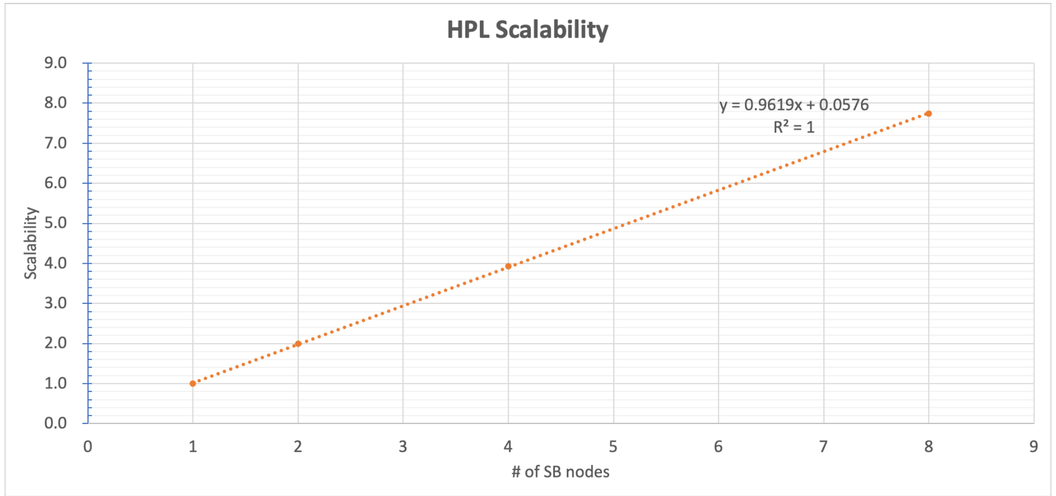
Workloads \ GPU Server Systems		System A *CPU:GPU (1:1)*	System B *CPU:GPU (1:2)*	System C *CPU:GPU (1:4)*
Scientific Simulations and Modeling	Computational Fluid Dynamics (CFD)	O/V	O	-/O
	Molecular Dynamics (MD)	V	O/V	O
	Numerical Weather Prediction (NWP)	O	-/O	-
	Computational Chemistry	V	O	O
	Quantum Mechanics / Physics	O/V	-/O	-
	Next-Generation Sequencing (NGS)	O	-/O	-
	Finite Element Analysis (FEA)	O	-/O	-
Image Processing and Visualization	Cloud Gaming	-	V	-/O
	3D Modeling and Computer-Aided Design (CAD)	O	V	-/O
	Video Transcoding and Streaming	-	V	-/O
	Virtual Reality (VR) and Augmented Reality (AR)	-	V	-
	Scientific Data Visualization and Analytics	-/O	V	-
	Medical Diagnostics, Imaging and Visualization	-/O	V	-/O
	Virtual Desktop Infrastructure (VDI)	-	V	-
	Video Analytics and Surveillance	-	V	-
Machine Learning and Artificial Intelligence	Image Classification and Object Detection	V	V	O/V
	Natural Language Processing (NLP) with Large Language Model (LLM)	O	-/O	V
	Virtual Assistants and Chatbot	-/O	-/O	-
	AI Generated Content (AIGC) with Generative Adversarial Networks (GAN)	O	O	V
	Recommendation Systems	O/V	O/V	-/O

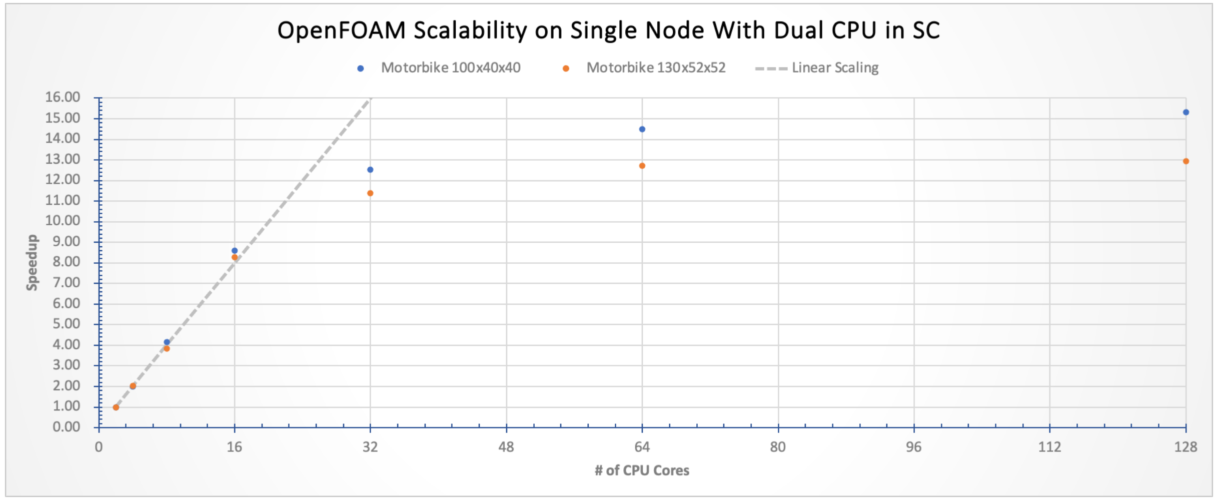
System A: *MBX-based with 1 GPU* | System B: *PCIe-based with up to 4 GPUs* | System C: *SXM5-based with up to 8 GPUs*

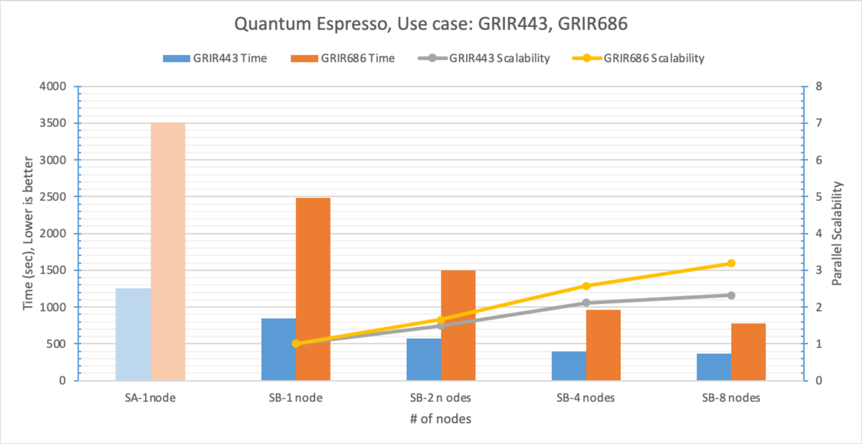
V: *Applicable* | O: *Conditional* | -: *Not Applicable*

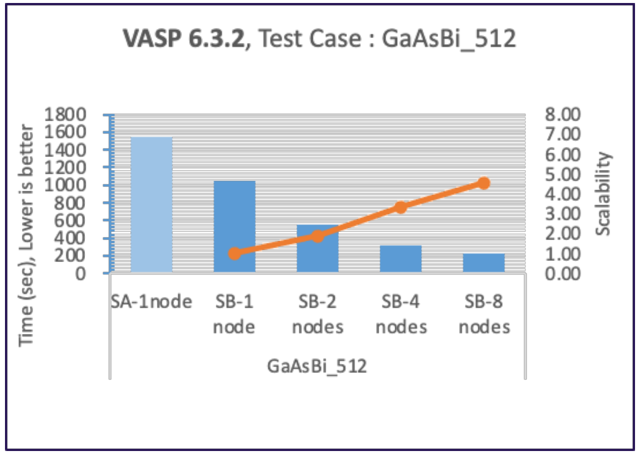
Scalability and Parallelism

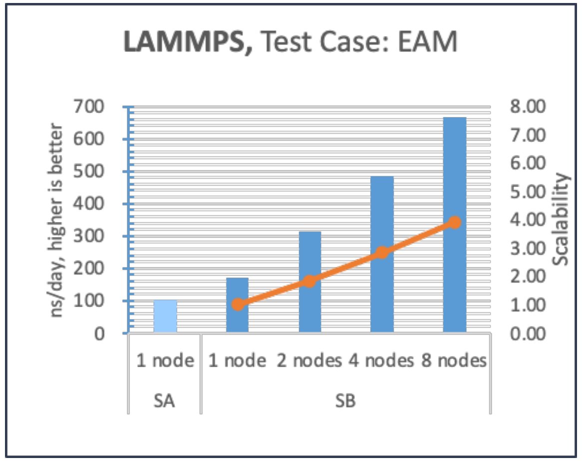
- Even when utilizing the same compute nodes within the same system, **scalability** can be influenced by **the characteristics, parallelism, and model parameters of different workloads and their datasets**
- BIOS settings and OS configuration in a system, network connectivity, and data I/O operations in storage also impact scalability in a cluster system.









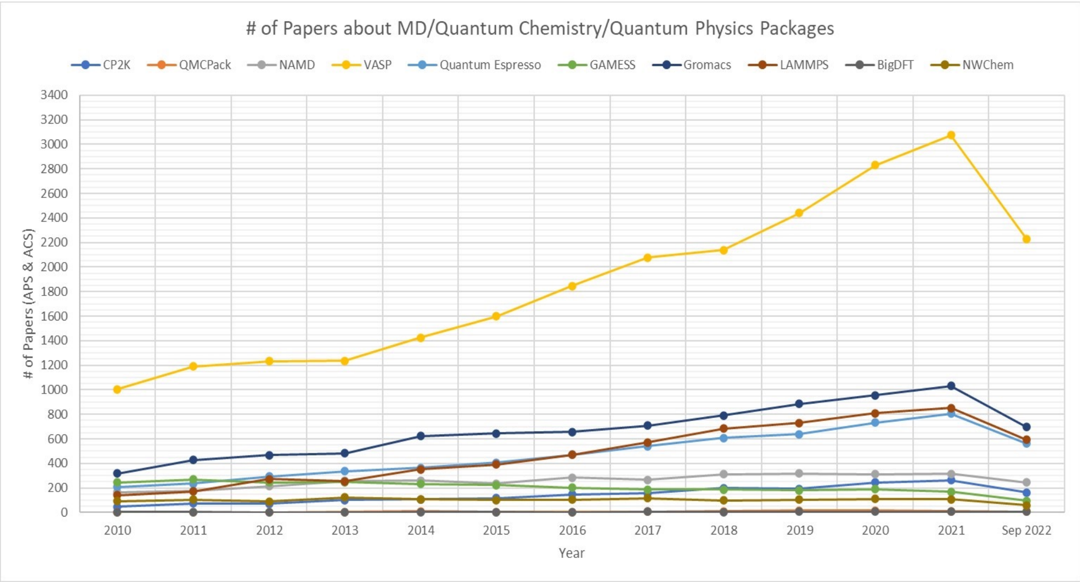


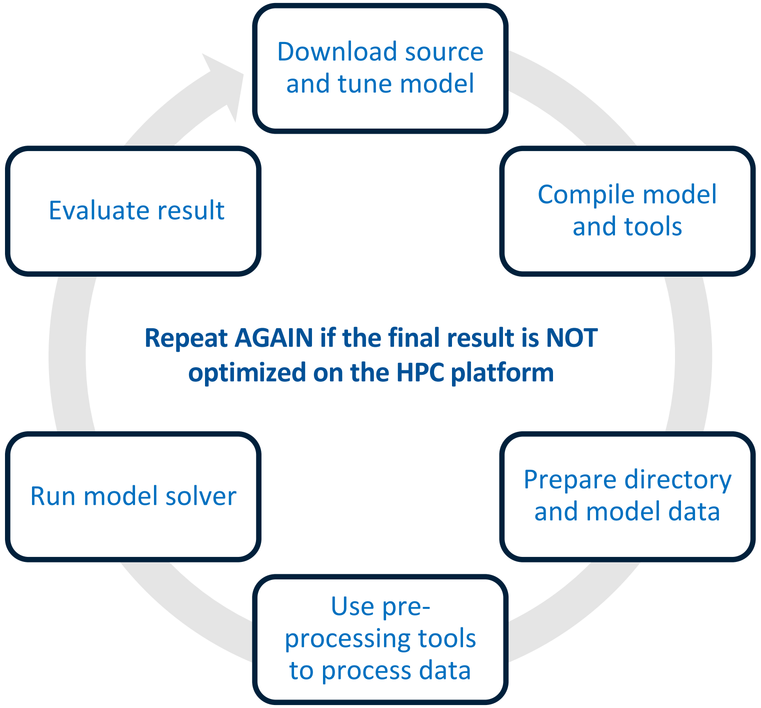
System Environment

To measure scalability, compare the execution of a workload on a single node in SA (System A) with SB (System B), and then expand SB by adding servers from 1 node, 2 nodes, up to 8 nodes.

Workload Analysis for Optimization

- Different workloads have different needs and the best-fit architecture based on their characteristics
- List several key HPC/AI applications used in vertical fields to optimize their performance with enabled CPU/GPU features
- Collaborate with vertical researchers, experts, ISV's and vendors to come out the BKC (Best Known Configuration) on system platforms with best practices in optimization to boost performance for vertical workloads





Workloads		Machine Learning		Data Analytics		High Performance Computing							
						Life Science		*Earth Exploration*		*FSI*		*Manufacturing*	
Category	Type	Training	Inference	Real Time	Batch	DNA Sequencing	Molecular Dynamics	Seismic Processing	Reservoir Engineering	Trading	Risk Mgmt.	EDA	CAE
Compute-bound	INT8 / INT16 / INT32	+	+	++	++					+	+		
	FP16 (HP)	++	+++	+	+								
	FP32 (SP)	+++	++	++	+++	++	+++			++	++		
	FP64 (DP)				+	+	+	+++	+++	+	+	+++	+++
Memory-bound	Shared	+++	+	++	+	+++	++	+++	++	+++	++	+	+++
	Distributed	+		+	++	+	+++	+	++	+	++	+++	+++
I/O-bound	Network	+++	++	+++	++	+	++	+	++	+++	+	+++	++
	Storage	+++		+	+++	+++	+	+	++	+	+++	+++	+

Performance Optimization

Discover
- What are the Constraints

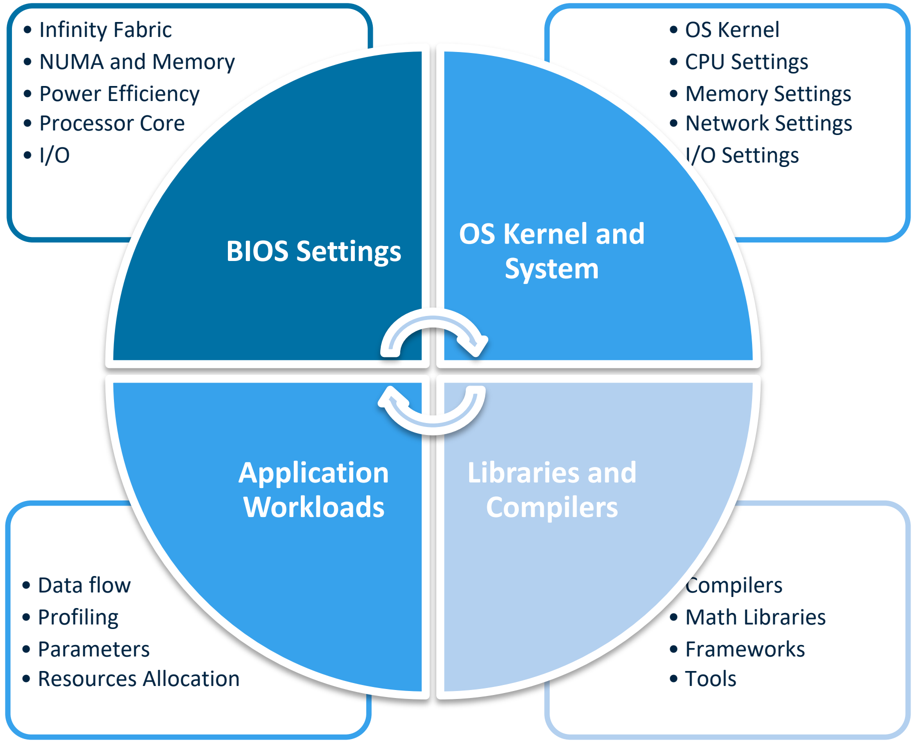
Design
- Theory & Execution

Measure
- Where is the App spending its time?

Refine
- Scientific Method Experiment

- **Considerations for Performance Optimization**
  - **Workloads Characteristics**
    - CPU Intensive / Memory Intensive / IO Intensive
    - Data pattern and workflow
  - **System Architecture**
    - Bare Metal to Virtualization / Containerization
    - Single node to Multi-nodes
    - Multi-cores to many-cores system
    - Heterogeneous computation between CPU and accelerators
    - Interconnections between nodes (bandwidth, latency)
    - I/O types and storage performance
  - **Software Stack and Programming Environment**
    - Combinations of libraries and compilers, compiling options, inter-connections, and workload component resources allocation

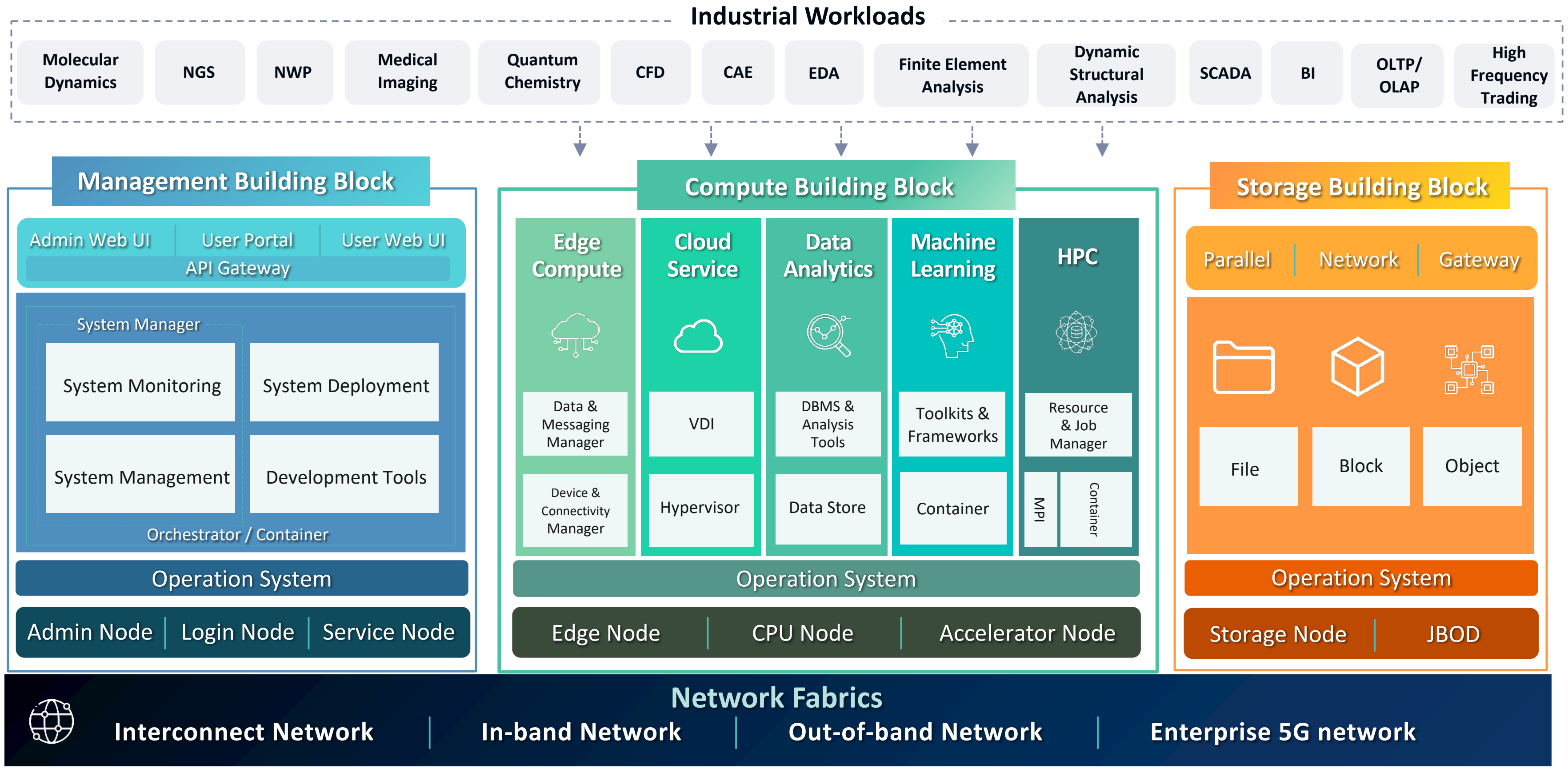
- **Approaches to optimize workload performance**
  - **Exploit pattern and data flow** - find the best-fit and deliverable architecture with the underlying hardware and infrastructure/middleware layers for major workloads
  - **Application performance management** - a set of monitoring and control tools enable users to tune their application environments
  - **Hardware/software pathing** - a vendor-driven effort that involves finding ways to enable a workload to move most expeditiously between middleware and infrastructure layers
  - **Tuned to the task** - matching a workload to the hardware platform best suited to serve
    - Reference Architecture
    - Best Practices



Workload-Driven Infrastructure for HPC/AI Workloads

www.qct.io

*QCT POD is a **Platform on Demand** concept, which provides a **pre-validated** and **pre-configurated on-premises** system with **best practice** software and hardware **integration for HPC/AI/DA workloads**.*



Developers

Provide a complete development platform and pre-compiled program modules to speed up the development process

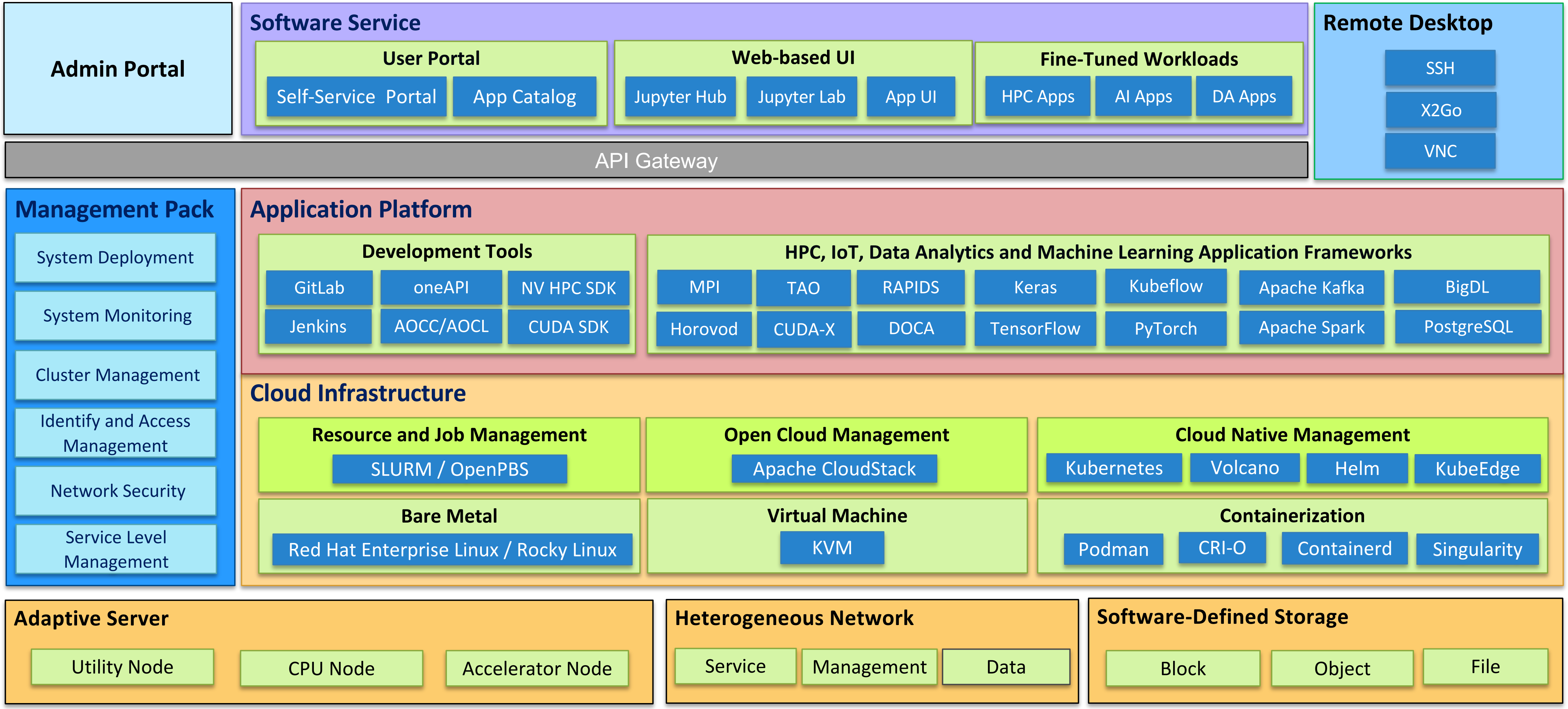
- **Robust Development Environment**
- **Fine-tuned Application Workloads**

Administrators

Through the system management module to simplify the deployment and management to improve the efficiency

- **Rapid System Deployment**
- **Realtime Monitoring**
- **Simplified Cluster Management**

Modern HPC/AI Workload-Driven System Architecture



Best Practices - QCT DevCloud Program

QCT DevCloud is a **comprehensive HPC/AI/DA environment** for user to **experience QCT POD solution and infrastructure expertise.** It includes **QCT precompiled workloads** and **development tool kits** support across a range of **QCT hardware platforms**, allowing end users to **remote access** and **test their applications** on a cluster environment.

- Heterogenous computing platform with HPC, AI, DA tool kits
- Cloud-native & Baremetal environment with resource and job management tools
- Cluster management and real-time monitoring
- Software defined storage with data tiering management

QCT precompiled workloads

Molecular Dynamics
GROMACS | LAMMPS | NAMD

Computational Fluid Dynamics
OpenFOAM

Quantum Chemistry
Quantum Espresso

Numeric Weather Prediction
WRF

Development toolkits

Computational Env.
JupyterHub

Frameworks
Pytorch | Tensorflow

Compiler &Libraries
Intel oneAPI | AMD AOCC/AOCL | Nvidia HPC SDK

Development Tools
Paraview | LMOD

Management tools

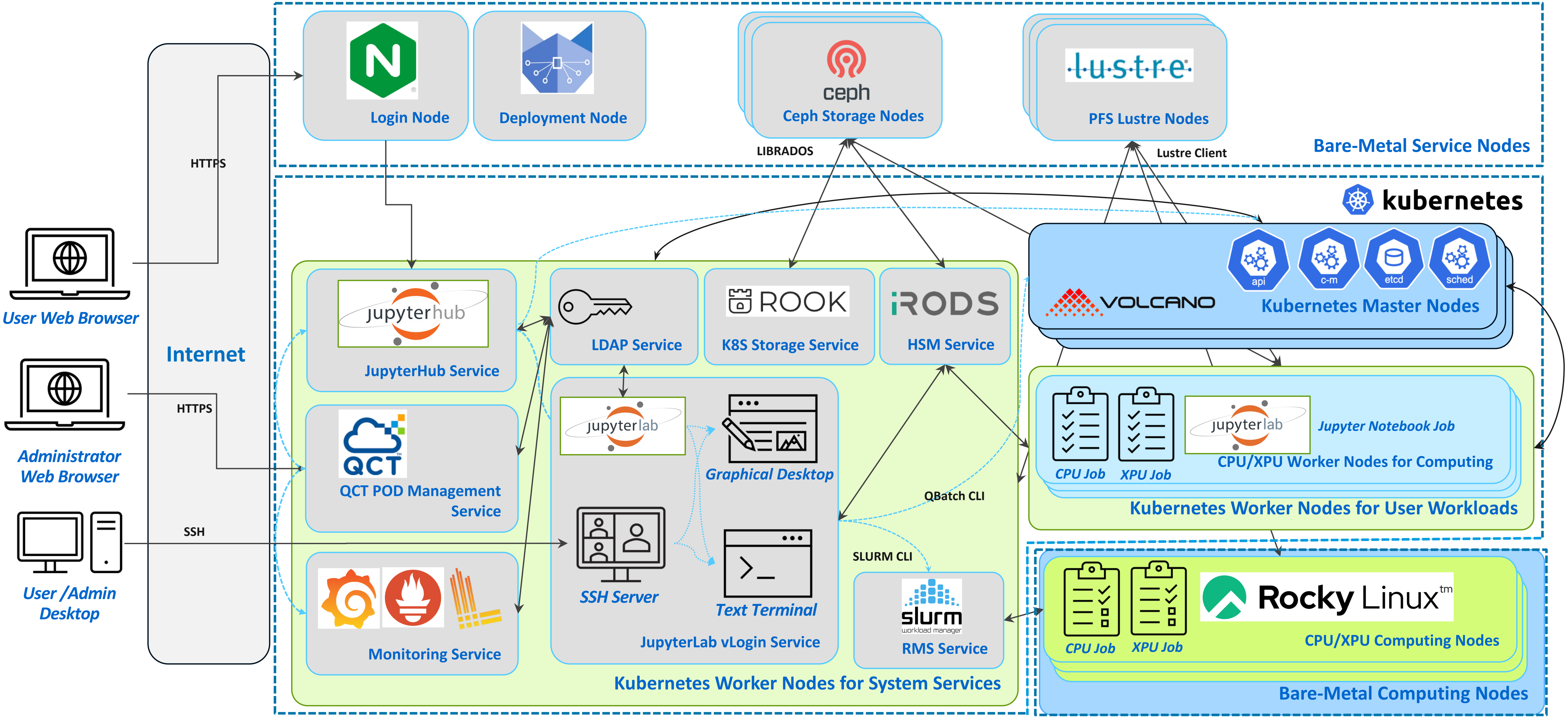
Storage system
iRODS | Ceph | Lustre

System Monitoring
Prometheus | Grafana

System Management
Volcano | SLURM | K8s | iRODS

Operating System
Rocky Linux

System Architecture for QCT DevCloud



The Value of QCT's Solution Offerings

Thank You

www.qct.io

5G AI