Empowered by Innovation NEC

NEC Hybrid Solutions for Meteo Sites

October 2nd, 2012

NEC Corporation

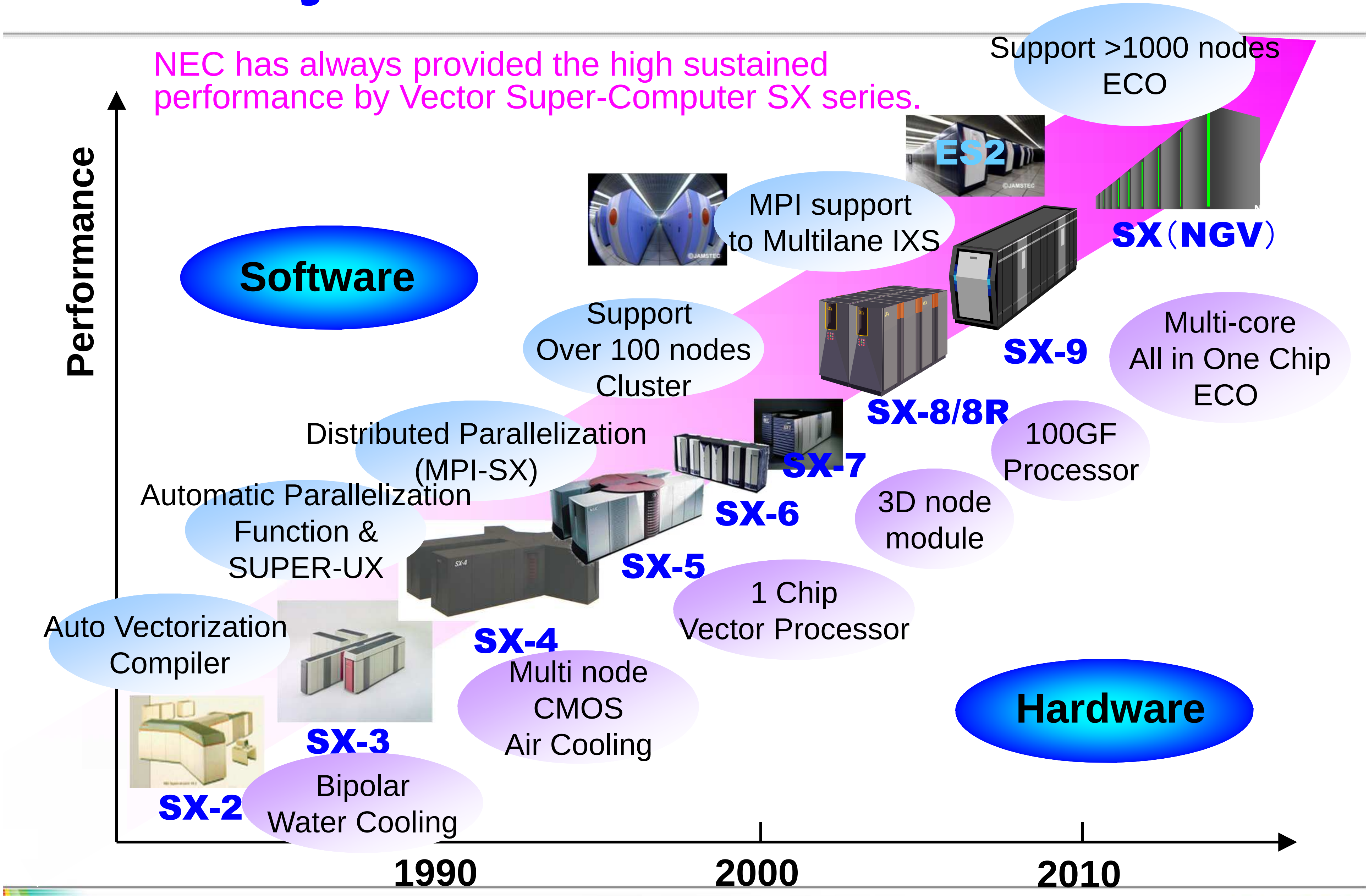
The contents of this presentation material is subject to change without notice.

Hybrid Concept, our strategy

- COTS(Commercial Off-The-Shelf) are adequate for quite some applications.
- But they are not the answer to every HPC-challenge.
- Consequently NEC will continue a proprietary vector architecture.
- The seamless integration of the vector-system with one build from standard components is the key of NEC’s strategy.
- In particular when complicated workflows need to be mapped on the best, i.e. most efficient hardware platform, as it is the case in production environments in the weather forecast business.

SX History and Technical evolutions

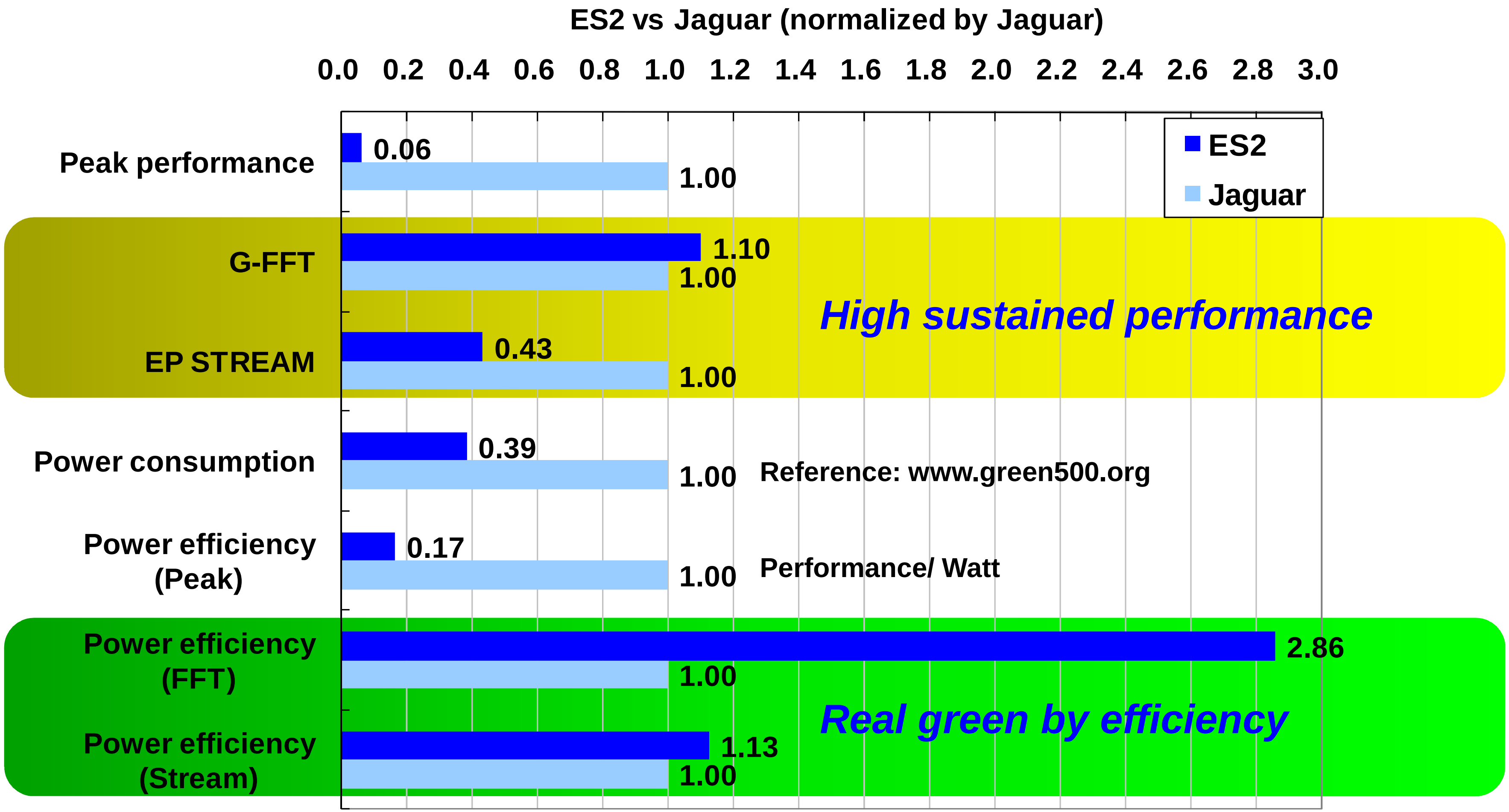
NEC has always provided the high sustained performance by Vector Super-Computer SX series.



NGV HARDWARE

Which is smarter ?

■Break the POWER WALL by “High computational efficiency”
■Higher sustained performance and efficiency are “SX DNA”



Targets of Next Generation Vector

High sustained performance

World’s top-class core performance (64GF)
World’s top-class memory bandwidth (64GB/s)

Inherit SX-DNA

Low power consumption

World’s top-class energy-saving supercomputer

compared to SX-9 $\frac{1}{10}$

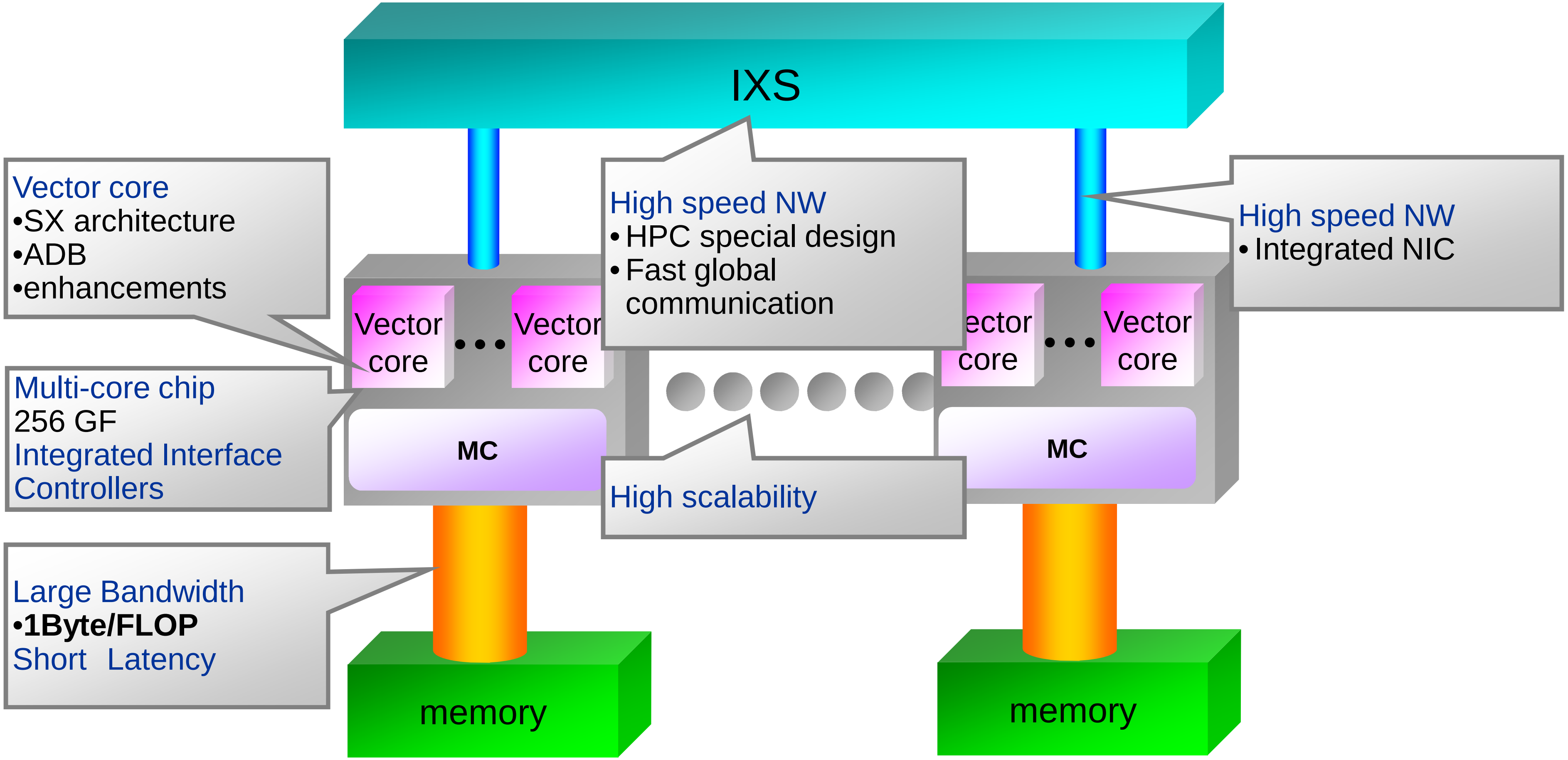
Small installation space

Reduce floor space cost

Compared to SX-9 $\frac{1}{5}$

Next Vector Configuration

The next generation multi-core vector architecture provides
high sustained performance at low power consumption



Vector core
- SX architecture
- ADB
- enhancements

Multi-core chip
256 GF
Integrated Interface Controllers

Large Bandwidth
- **1Byte/FLOP**

Short Latency

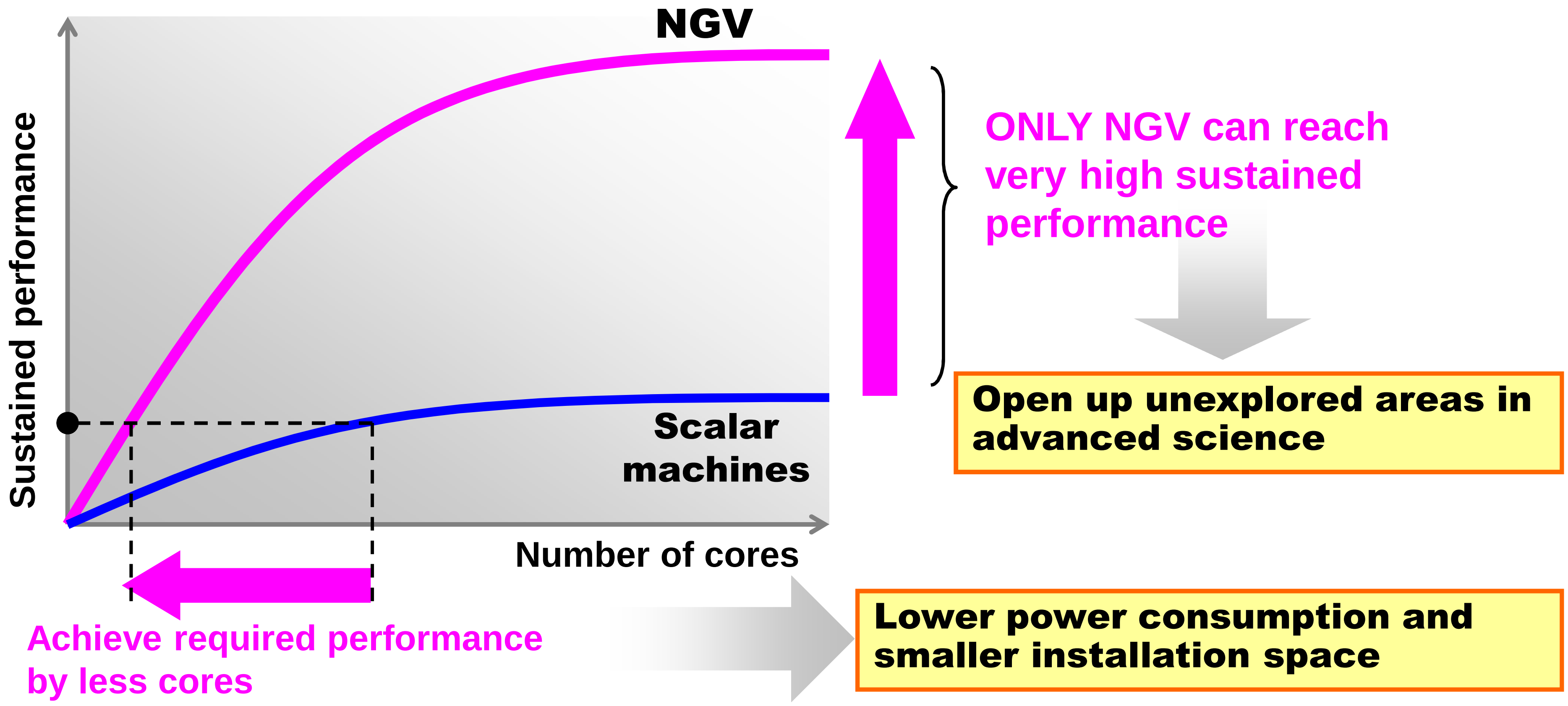
High speed NW
- HPC special design
- Fast global communication

High scalability

High speed NW
- Integrated NIC

Higher Sustained Performance by Powerful Core

- Very high system SUSTAINED performance = SX DNA
  - →required performance with less cores
  - → parallel efficiency
  - → lower power and space
- "Green HPC"



ONLY NGV can reach very high sustained performance

Open up unexplored areas in advanced science

Achieve required performance by less cores

Lower power consumption and smaller installation space

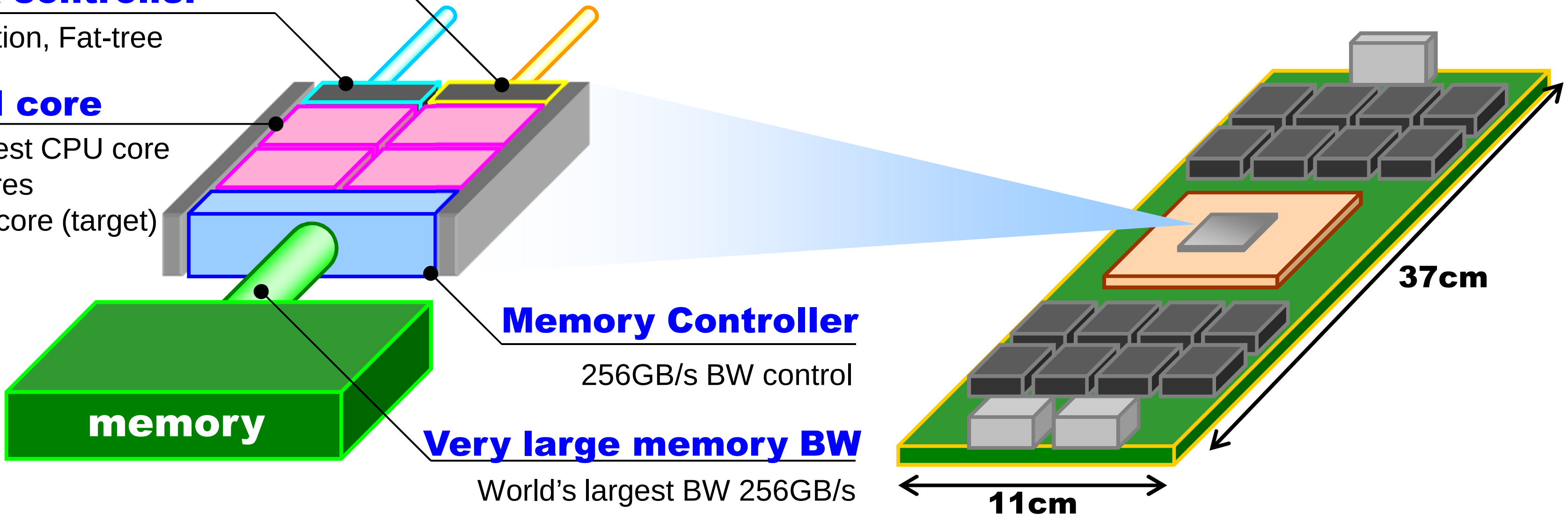
All-in-one Processor

- 4 powerful cores and each interface controllers are integrated in one-CPU → Power saving
- Compact card design → Space saving

NGV CPU

CPU card

Performance: 256GF
Memory bandwidth: 256GB/s



I/O Controller
Connection to storage device, Ethernet

Network controller
8GB/s/direction, Fat-tree

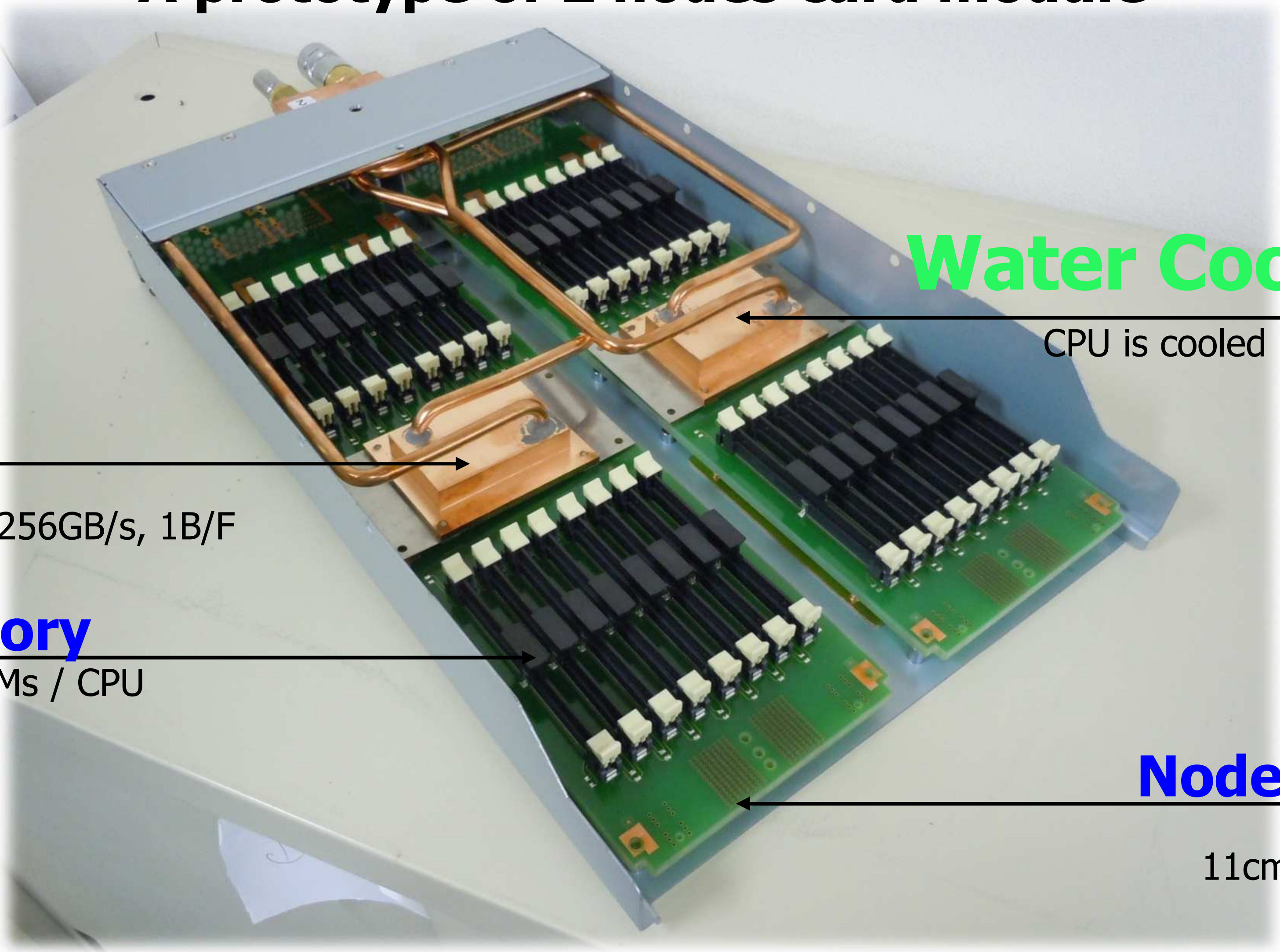
Powerful core
World's fastest CPU core
64GF x 4cores
1MB cache/core (target)

Memory Controller
256GB/s BW control

Very large memory BW
World's largest BW 256GB/s

Packaging

A prototype of 2 nodes card module



Water Cooling

CPU is cooled by water

CPU

4 cores
256GF, 256GB/s, 1B/F

Memory

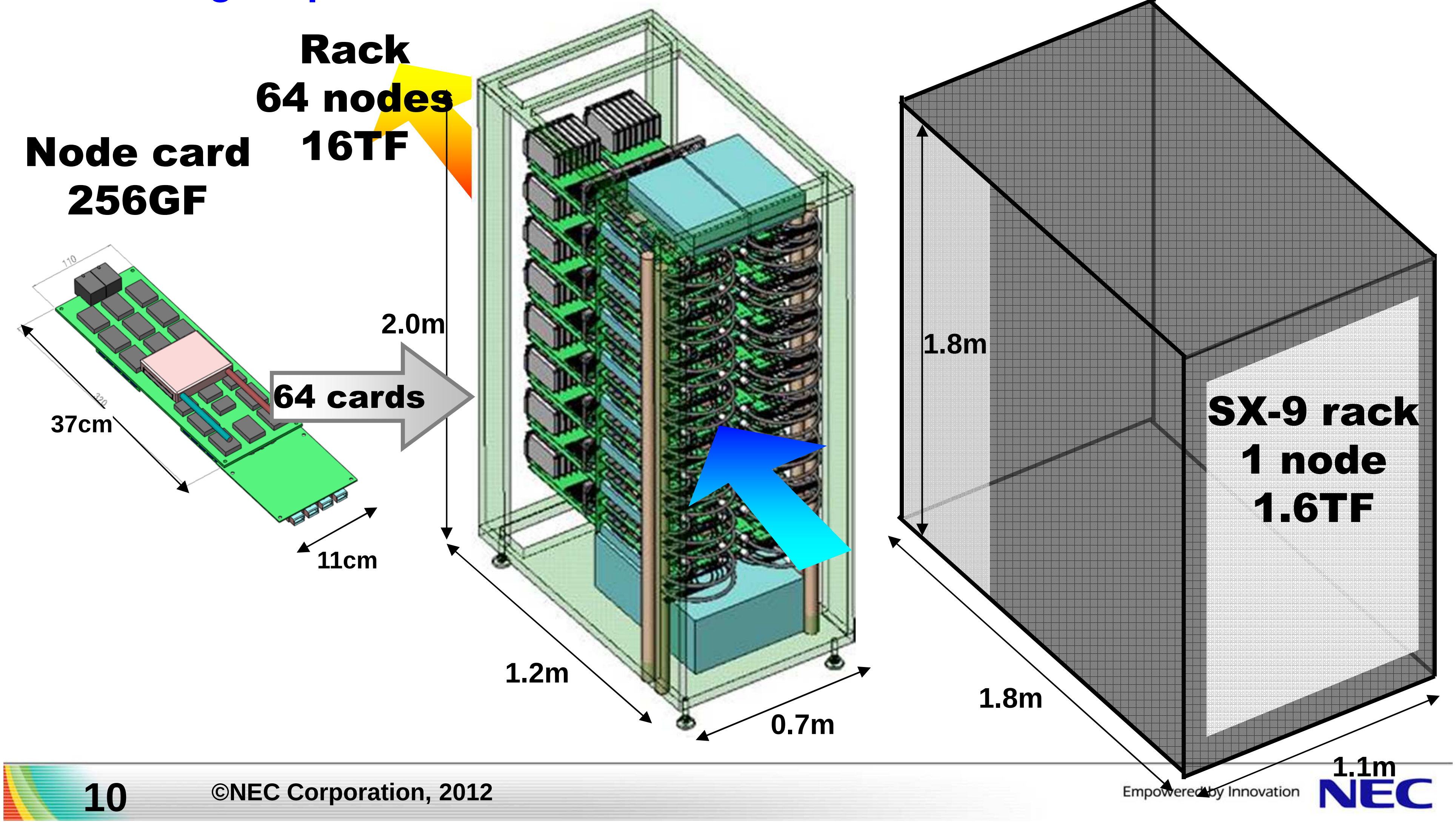
16 DIMMs / CPU
64GB

Node card

1 CPU
11cm x 37cm

Rack Implementation

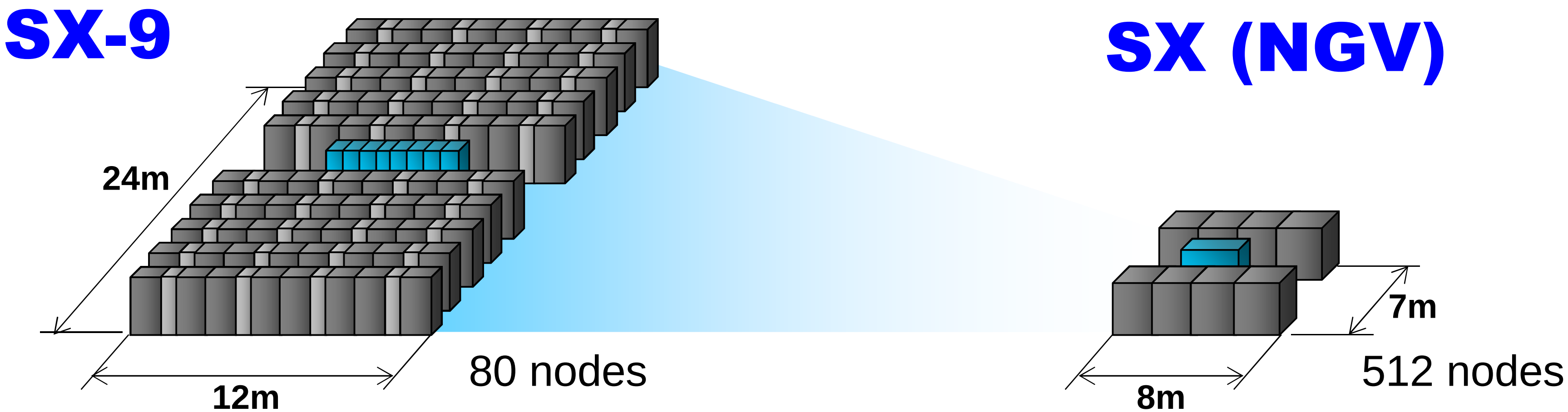
- ■64 nodes are implemented in one rack = 16TF
- ■hybrid cooling by air and liquid
- ■10x higher performance and over 2x smaller rack vs. SX-9



Downsizing and Power Saving

Providing 5x smaller space and 10x lower power consumption compared to SX9 by GREEN design and compact implementation.

Comparison with same performance (131TF)



25m pool size

Meeting room size

131TF
288m²
2.4MW

space **1/5**
power **1/10**

131TF
56m²
0.24MW

NGV SOFTWARE

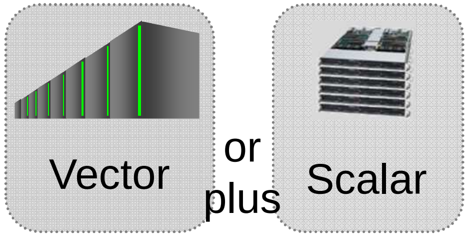
NGV Software Overview

Provide hybrid cluster solution

- Integrate vector and scalar cluster as a single system -

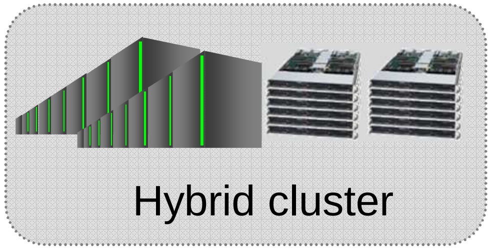
Background

- ■Demanding computation power for HPC applications
- ■Not one kind of architecture will fulfill all requirements



Solution

- ■Provide hybrid solution
  - ● Job collaboration using workflow tools
  - ● Integrated scheduling (assign right node to right job)
  - ● New shared file system
- ■Provide large cluster solution
  - ● Integrated single system management of vector and scalar cluster
  - ● Enhanced scalability and reliability
- ■And much more
  - ● Sophisticated OS and compiler compliant with standards
  - ● MPI-3 support, enhanced performance (memory and interconnect)
  - ● User-friendly tools, easy-to-use debugging environment, etc.



Integrated scheduler (Batch system)

Shared File System

OS & Compiler

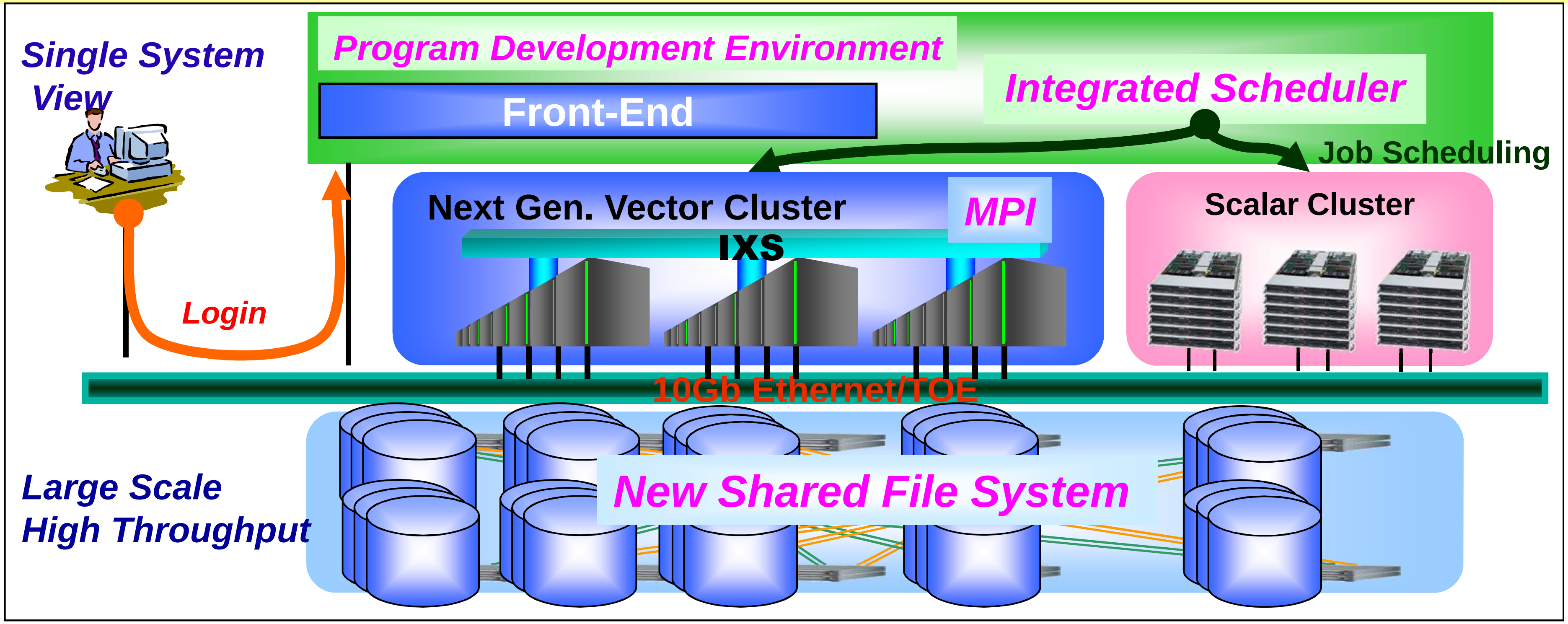
MPI

Tools

System Overview

■Single system solution – Integrated scheduler –

- ●Supports vector clusters and scalar clusters together as a single system
- ●Easy to manage a system with more than 1000 nodes

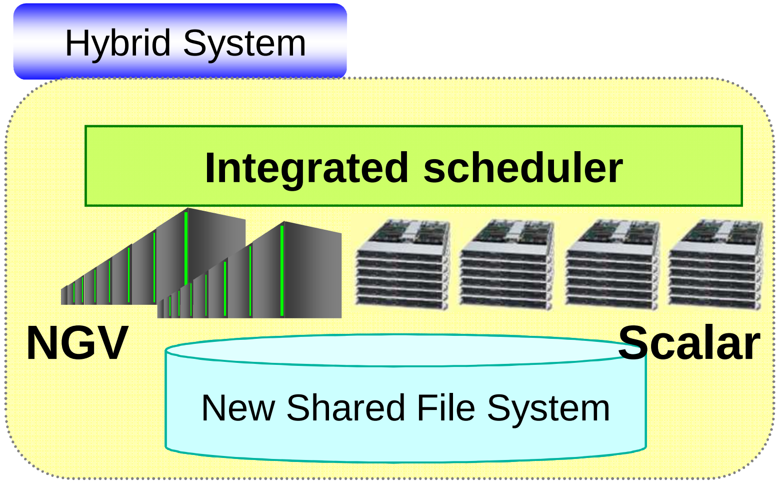


■New I/O solution

- ●Realizes new shared file system with huge capacity and large scale using multiple IO servers
- ●Provides high speed IO using proprietary protocol lighter than NFS

Integrated Scheduler

- Integrated scheduler realizes enhanced hybrid system running real **workflow!**
  - ●Vector cluster and scalar cluster managed as a single system
  - ●Collaboration scheduling of vector jobs and scalar jobs using a **workflow script**



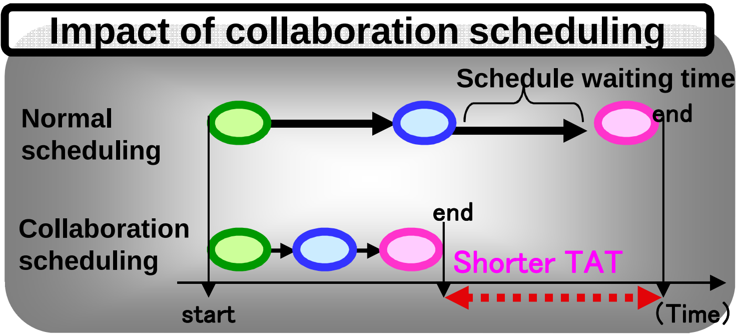
Vector and scalar system closely coupled

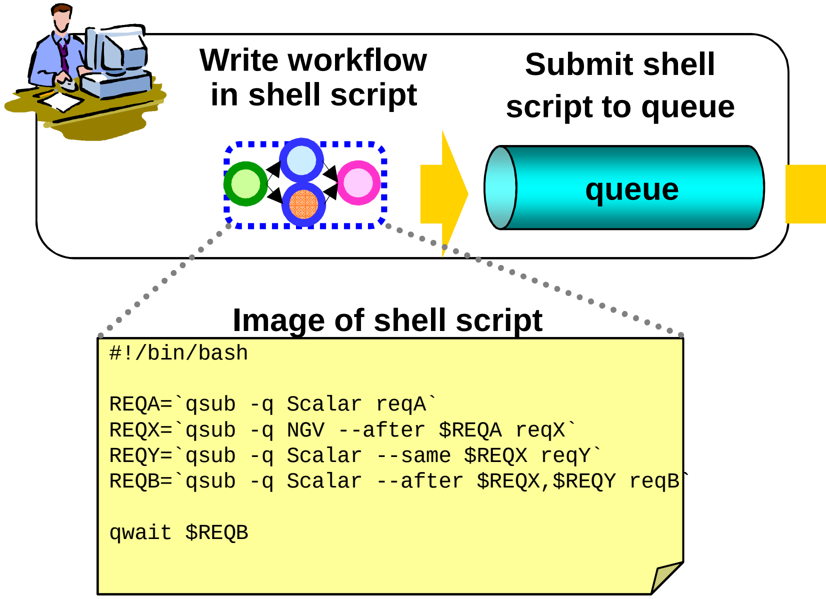
- Easy operation of large scale cluster system
  - ●Enhanced scalability
  - ●Inter cluster scheduling
  - ●Ensemble job (parameter sweep)

Job collaboration realizes seamless usage of hybrid system

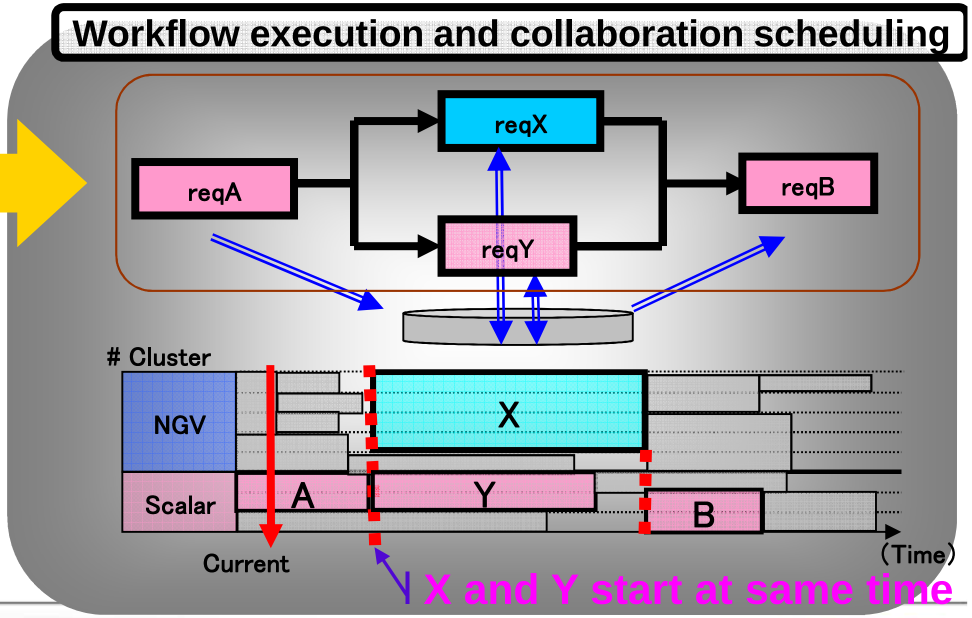
Efficient execution of Job collaboration

- Job execution order can be specified by workflow tools
  - Serial/Parallel execution
  - Conditional branch by exit code
- Collaboration scheduling of vector and scalar jobs
  - Collaboration jobs to be executed consecutively to shorten TAT





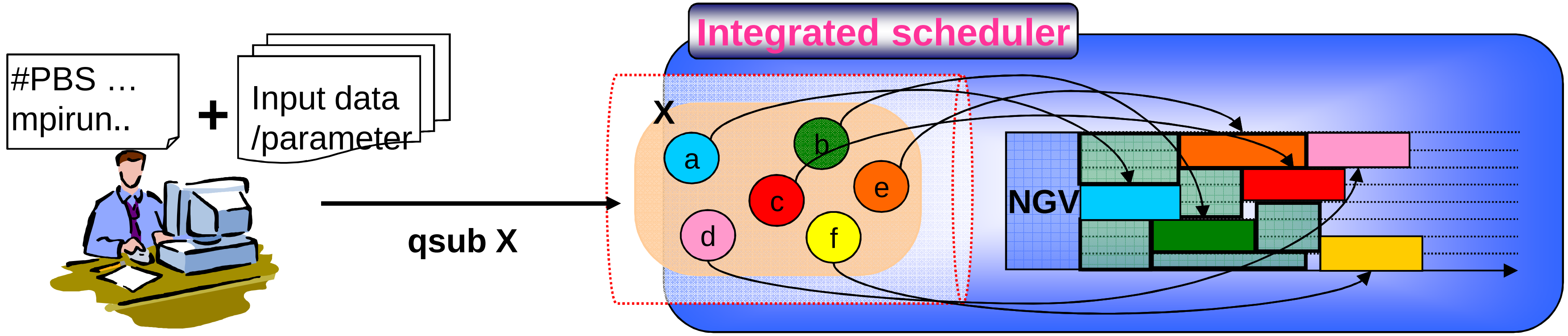
```
#!/bin/bash

REQA=`qsub -q Scalar reqA`
REQX=`qsub -q NGV --after $REQA reqX`
REQY=`qsub -q Scalar --same $REQX reqY`
REQB=`qsub -q Scalar --after $REQX,$REQY reqB`

qwait $REQB
```


Workflow execution and collaboration scheduling
reqX
reqA
reqB
reqY
# Cluster
NGV
Scalar
X
A
Y
B
Current
(Time)
X and Y start at same time

Ensemble job supported

Run same job with many different parameters (parameter sweep)

- Submit once and thousands of jobs are scheduled immediately
- Sub-job for each input file is created automatically



Submit once (Collective qsub)

and

All sub-jobs are scheduled at once

- Collective qdel, specific qdel, etc. are also supported

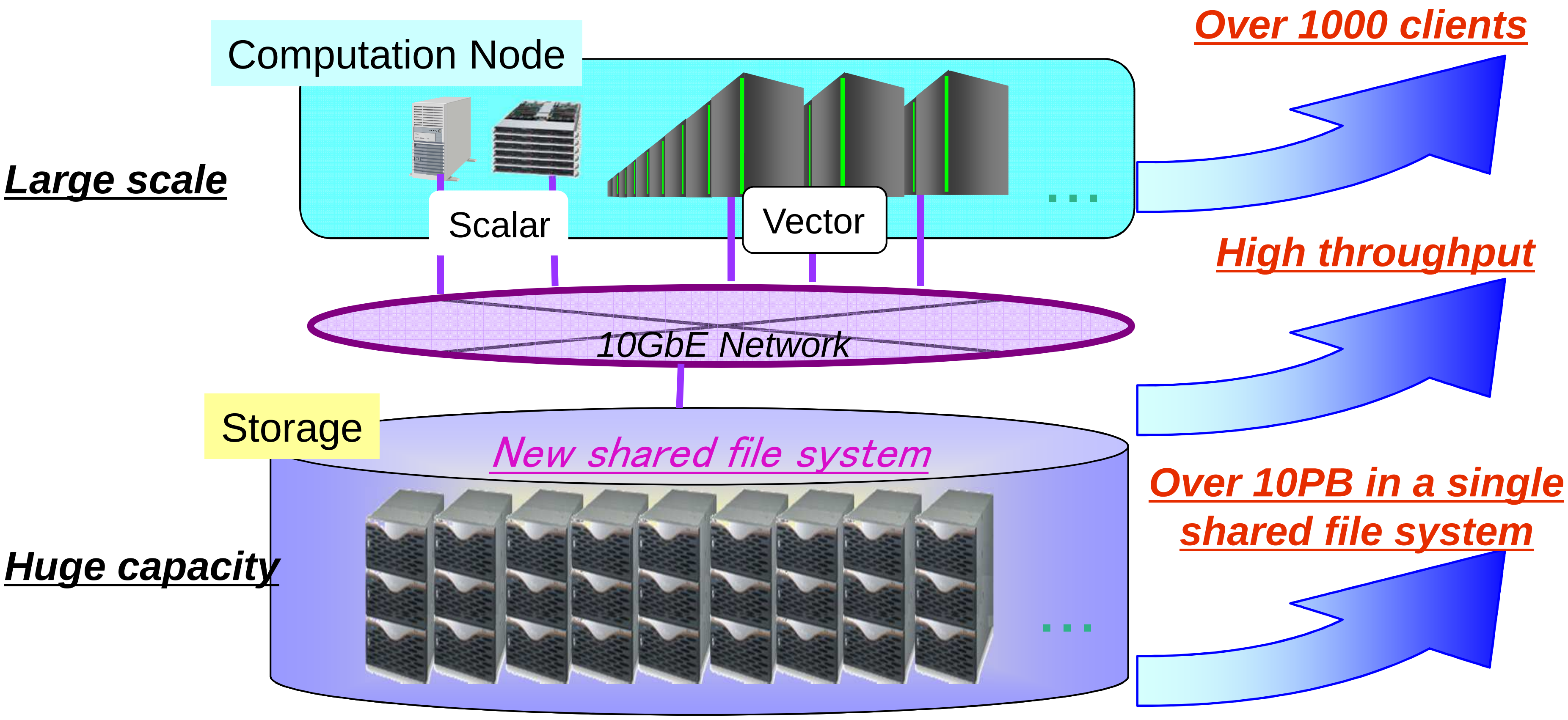
```
Ex) qdel X : delete all sub-jobs
    qdel X.a : delete sub-job X.a
```

- Convenient parameter generation features
  - Sequential data generation (sub-job number, date, time, etc.)
  - Generate parameter from listed filename, etc.

```
Ex) IN-DATA.%(date:0530-0601)
    -> IN-DATA.0530
       IN-DATA.0531
       IN-DATA.0601
```

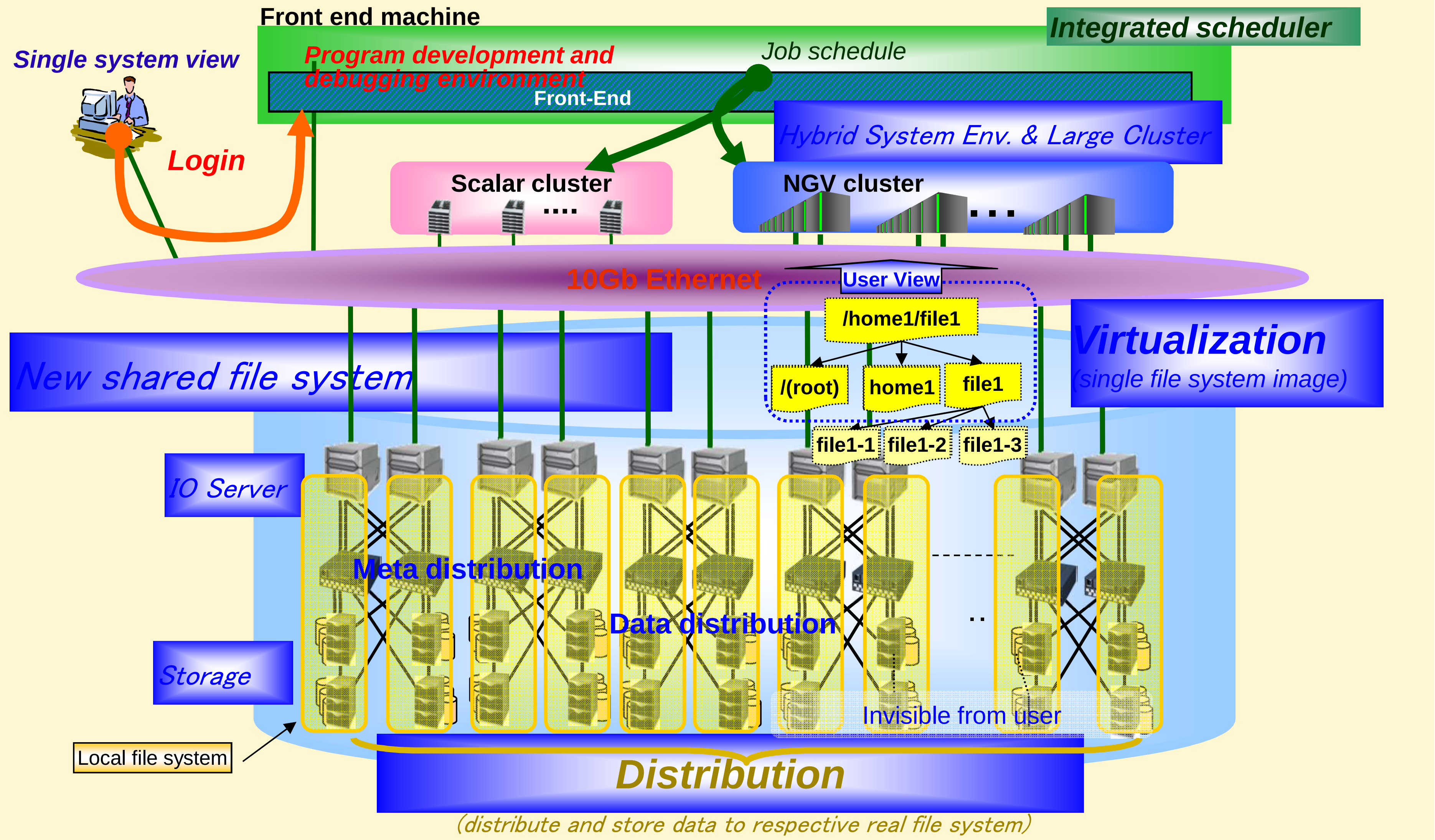
New shared file system

New shared file system provides fast I/O to massive data



File system image

- Provide single file system view for large scale cluster



Performance improvement - key points -

Light proprietary I/O protocol

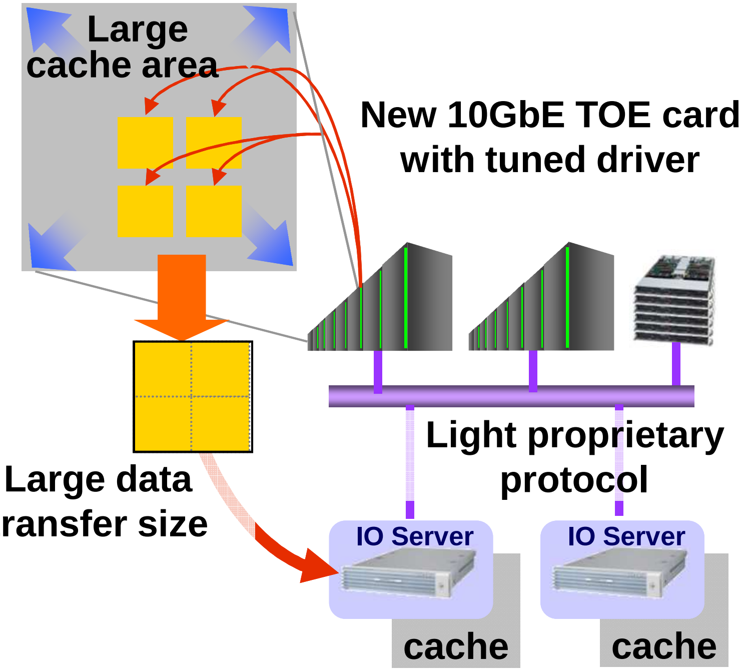
- Efficient data transfer between server and client (Gather data into single request and send together)
- Adopt next generation 10GbE TOE card and optimize network driver for the new card

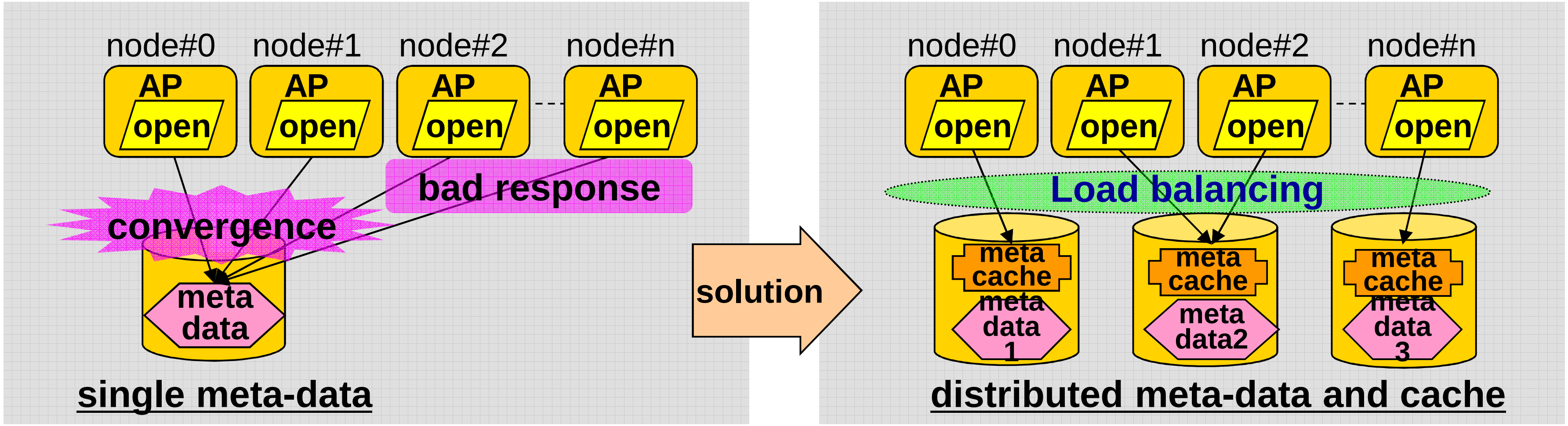
Data cache

- Efficient IO handling using large data cache on client and IO servers.

Avoid congestion on meta-data server!

- **Meta-data distribution,** not single MDS like Lustre
- Reduce network traffic using meta-data cache



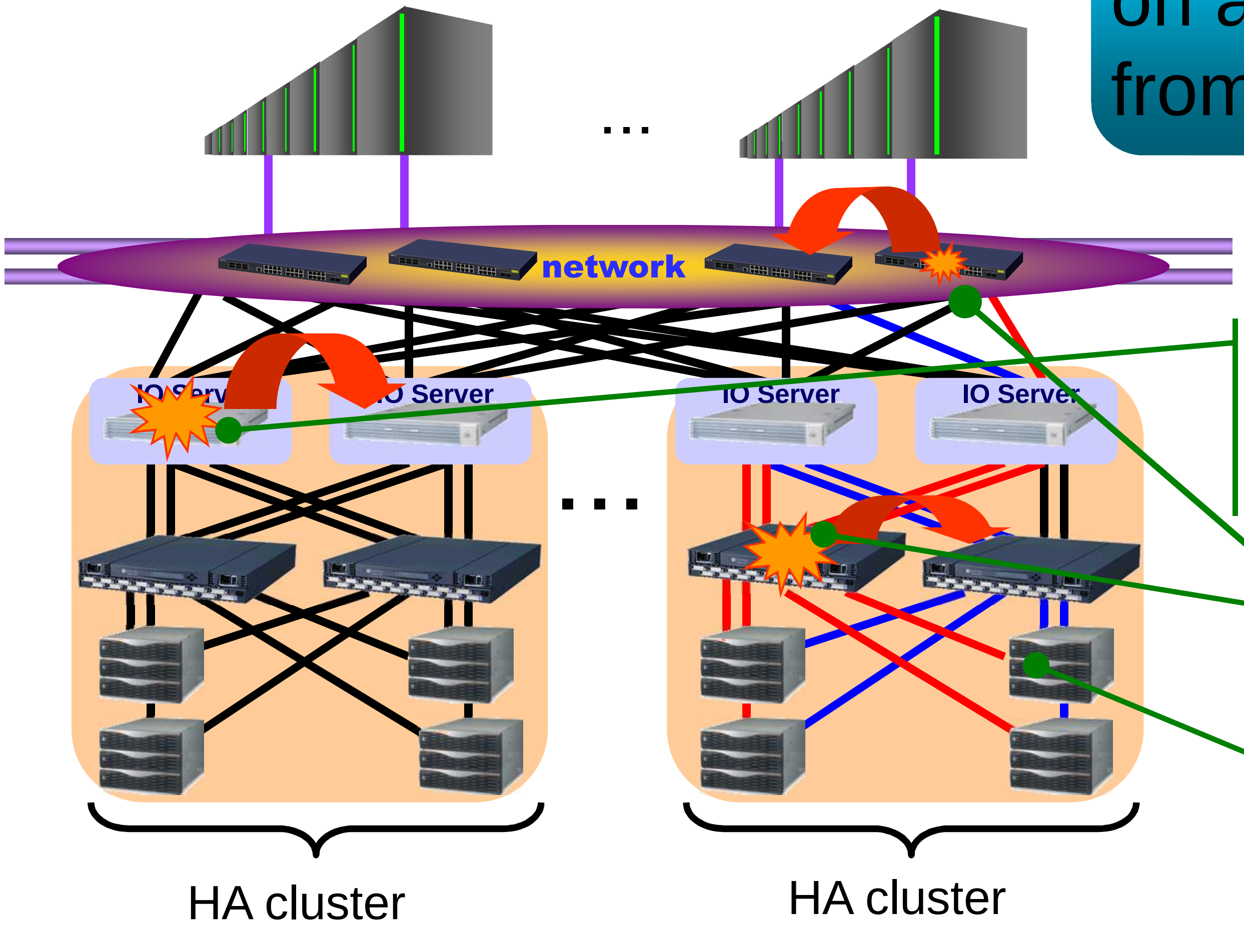


Reliability

Responses to system failure
(Realization of high availability)

- System continuity and data preservation

Possible redundancy on all paths from node to storage



(1) Server failure
- Server failover

Servers are configured Active-Active or Active-Standby

(2) Path failure
- Path failover

(3) Disk failure
- RAID5, RAID6

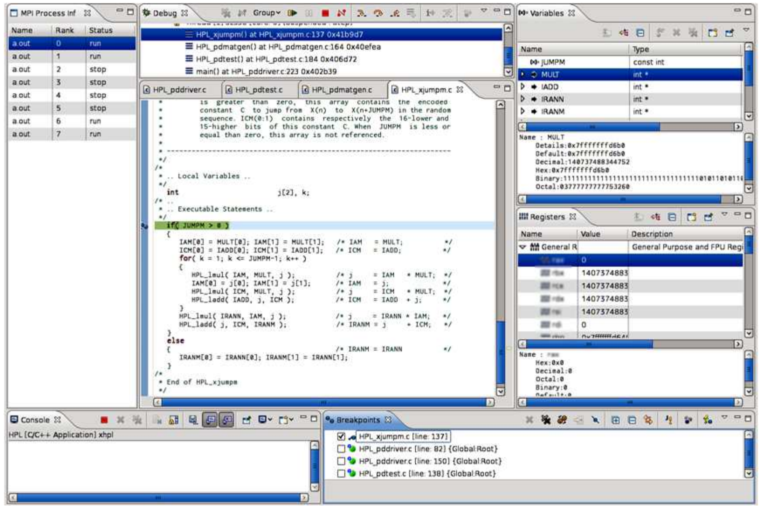
And much more ...

Fortran 2003, C/C++ compilers and MPI

- ISO standard compliant
- Sophisticated automatic vectorization and parallelization
- Sophisticated usage of ADB (~ vector cache)
- Automatic optimization and optimization by directives
- OpenMP and MPI-3 support

GUI Tools and debuggers

- GUI Performance analysis/tuning tool
- GUI debugger



Name	Rank	Status
a.out	0	run
a.out	1	run
a.out	2	stop
a.out	3	stop
a.out	4	stop
a.out	5	stop
a.out	6	run
a.out	7	run

```
HPL_xjumpm() at HPL_xjumpm.c:137 0x41b9d7
HPL_pdmatgen() at HPL_pdmatgen.c:164 0x40efea
HPL_pdtest() at HPL_pdtest.c:184 0x406d72
main() at HPL_pddriver.c:223 0x402b39
```

```
 *          is  greater  than  zero,  this  array  contains  the  encoded
 *          constant  C  to jump from  X(n)  to  X(n+JUMPM) in the random
 *          sequence. ICM(0:1)  contains  respectively  the  16-lower  and
 *          15-higher  bits  of this constant  C. When  JUMPM  is less or
 *          equal than zero, this array is not referenced.
 *
 * ---------------------------------------------------------------------
 */
/*
 * .. Local Variables ..
 */
   int                        j[2], k;
/* ..
 * .. Executable Statements ..
 */
   if( JUMPM > 0 )
   {
      IAM[0] = MULT[0]; IAM[1] = MULT[1];   /* IAM   = MULT;          */
      ICM[0] = IADD[0]; ICM[1] = IADD[1];   /* ICM   = IADD;          */
      for( k = 1; k <= JUMPM-1; k++ )
      {
         HPL_lmul( IAM, MULT, j );          /* j     = IAM   * MULT;  */
         IAM[0] = j[0]; IAM[1] = j[1];      /* IAM   = j;             */
         HPL_lmul( ICM, MULT, j );          /* j     = ICM   * MULT;  */
         HPL_ladd( IADD, j, ICM );          /* ICM   = IADD  + j;     */
      }
      HPL_lmul( IRANN, IAM, j );            /* j     = IRANN * IAM;   */
      HPL_ladd( j, ICM, IRANM );            /* IRANM = j     * ICM;   */
   }
   else
   {                                        /* IRANM = IRANN          */
      IRANM[0] = IRANN[0]; IRANM[1] = IRANN[1];
   }
/*
 * End of HPL_xjumpm
 */
```

Name	Type
JUMPM	const int
MULT	int *
IADD	int *
IRANN	int *
IRANM	int *

```
Name : MULT
    Details:0x7fffffffd6b0
    Default:0x7fffffffd6b0
    Decimal:140737488344752
    Hex:0x7fffffffd6b0
    Binary:111111111111111111111111111111111110101101011
    Octal:03777777777753260
```

Name	Value	Description
General R		General Purpose and FPU Regi
[illegible]	0	
[illegible]	1407374883	
[illegible]	1407374883	
[illegible]	1407374883	
[illegible]	1407374883	
[illegible]	0	

```
Name :
    Hex:0x0
    Decimal:0
    Octal:0
    Binary:0
```

HPL [C/C++ Application] xhpl

- ☑ HPL_xjumpm.c [line: 137]
- □ HPL_pddriver.c [line: 82] {Global:Root}
- □ HPL_pddriver.c [line: 150] {Global:Root}
- □ HPL_pdtest.c [line: 138] {Global:Root}

All software are tuned and optimized to extract best performance out of NGV

Empowered by Innovation

NEC