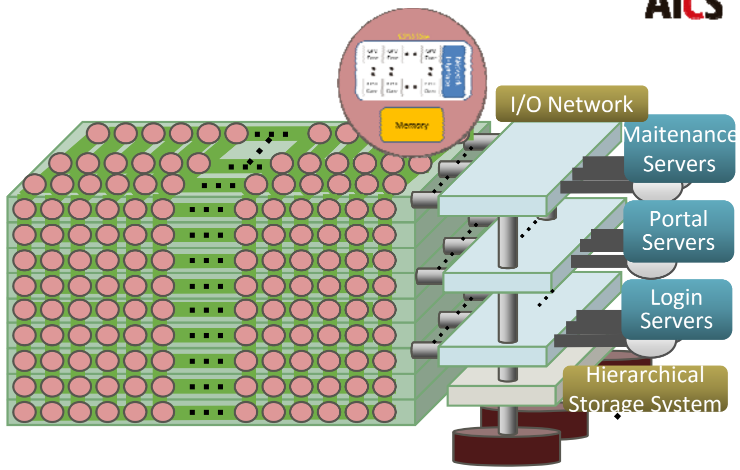
Development of system software stack for post K computer

Yutaka Ishikawa
RIKEN AICS

FLAGSHIP2020 Project



☐ Missions

- Building the Japanese national flagship supercomputer, post K, and
- Developing wide range of HPC applications, running on post K, in order to solve social and science issues in Japan

☐ Hardware and System Software

- Post K Computer
  - RIKEN AICS is in charge of development
  - Fujitsu is vendor partnership

☐ Applications

- 9 High priority issues from a social and national viewpoint
  - Promising creation of world-Leading achievement
  - Promising strategic use of post K computer

CY	2014	2015	2016	2017	2018	2019	2020
	Q1 Q2 Q3 Q4	Q1 Q2 Q3 Q4	Q1 Q2 Q3 Q4	Q1 Q2 Q3 Q4	Q1 Q2 Q3 Q4	Q1 Q2 Q3 Q4	Q1 Q2 Q3 Q4

Basic Design | Design and Implementation | Manufacturing, Installation, and Tuning | Operation

An Overview of Architecture

Node and Storage Architecture

- Manycore architecture
- 3 level hierarchical storage system
  - Silicon Disk
  - Magnetic Disk
  - Storage for archive

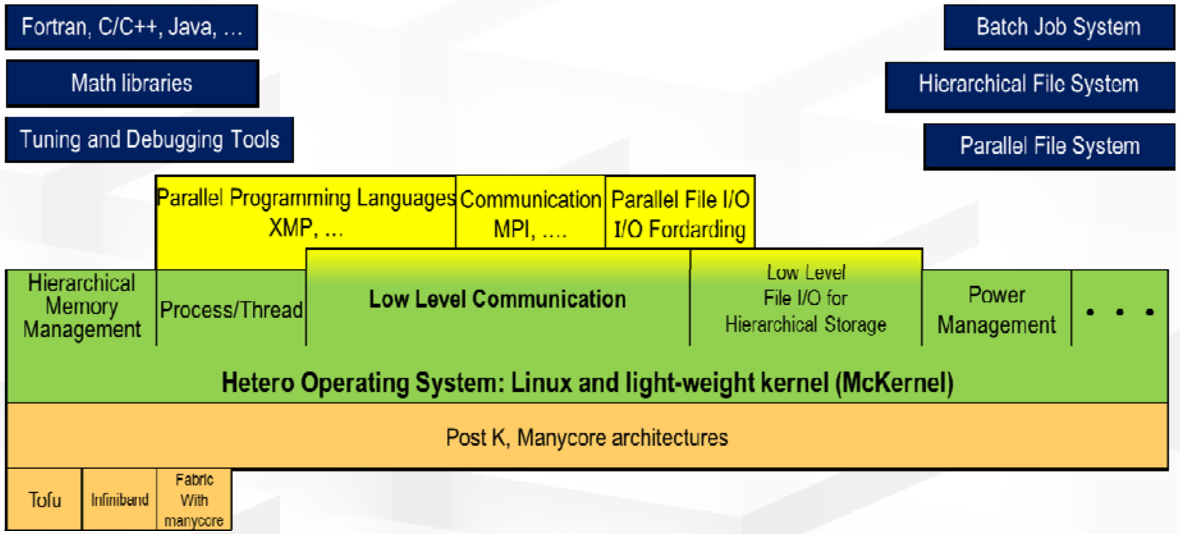
Cf. Fujitsu Latest Supercomputer FX100

Announced in 2014, and now delivery

Architecture	SPARCV9 + HPC-ACE2
No. of cores	32 compute cores + 2 assistant cores
Peak performance	1+ TF
Memory	32 GB (HMC) read: 240 GB/s, write: 240 GB/s
Interconnect	Tofu2: 12.5 GB/s x 2 (bidirection) x 10 link

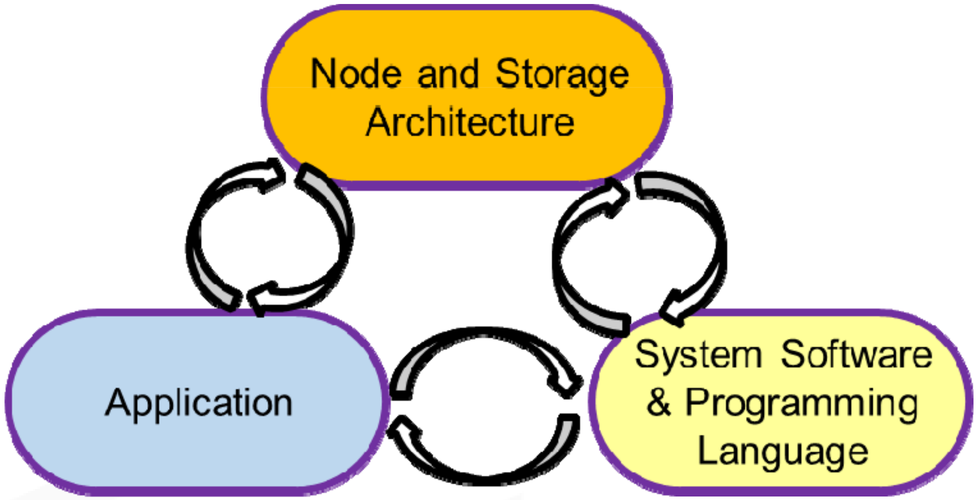
http://www.fujitsu.com/global/Images/primehpc-fx100-datasheet-en.pdf

System Software Architecture



Design Philosophy in System Software Stack

1. **Lesson Learned from K computer development and operation**
2. **Codesign Trinity**

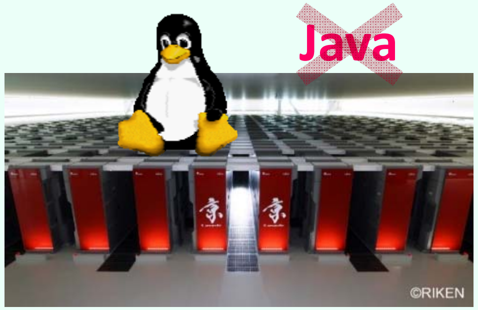


3. **Separation of Mechanism and Policy**

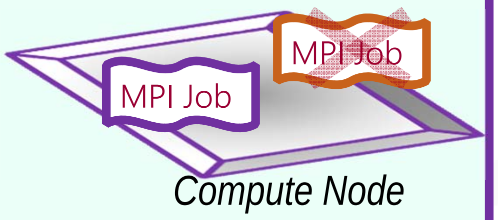
Lesson Learned from K development and operation

- **Much application users involvement**
- **Usability & Flexibility, File System, RAS capabilities**

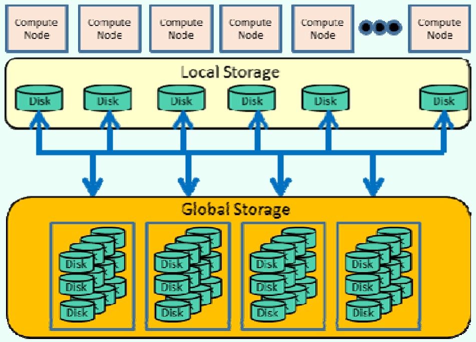
Examples:



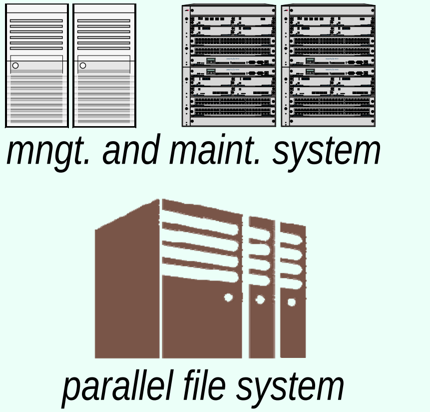
Though Linux runs on compute nodes, not all languages/tools available in x86 Linux are provided



Only single MPI job may run on each node. This is not an operation policy, but the implementation limitation



File staging causes unnecessary files movement. It seems that users do not specify exact required files



Reviewing equipment redundancy and system software stack, especially parallel file system, for RAS

An Overview of Co-design in the Post K development

☐ Node and Storage Architecture

- #SIMD, SIMD length, #core, #NUMA node
- cache (size and bandwidth)
- network (topologies, latency and bandwidth)
- memory technologies
- specialized hardware
- Node interconnect, I/O network

☐ System Software

- Operating system for many core architecture
- Communication libraries (low level layer, MPI, PGAS)
- File I/O (Asynchronous I/O, buffering/caching)

☐ Programming Environment

- Programming model and languages
- Math libraries, domain-specific libraries

9 social & scientific priority issues and their R&D organizations have been selected from the following point of view:

- High priority issues from a social and national viewpoint
- Promising creation of world-Leading achievement
- Promising strategic use of post K computer

Target Application

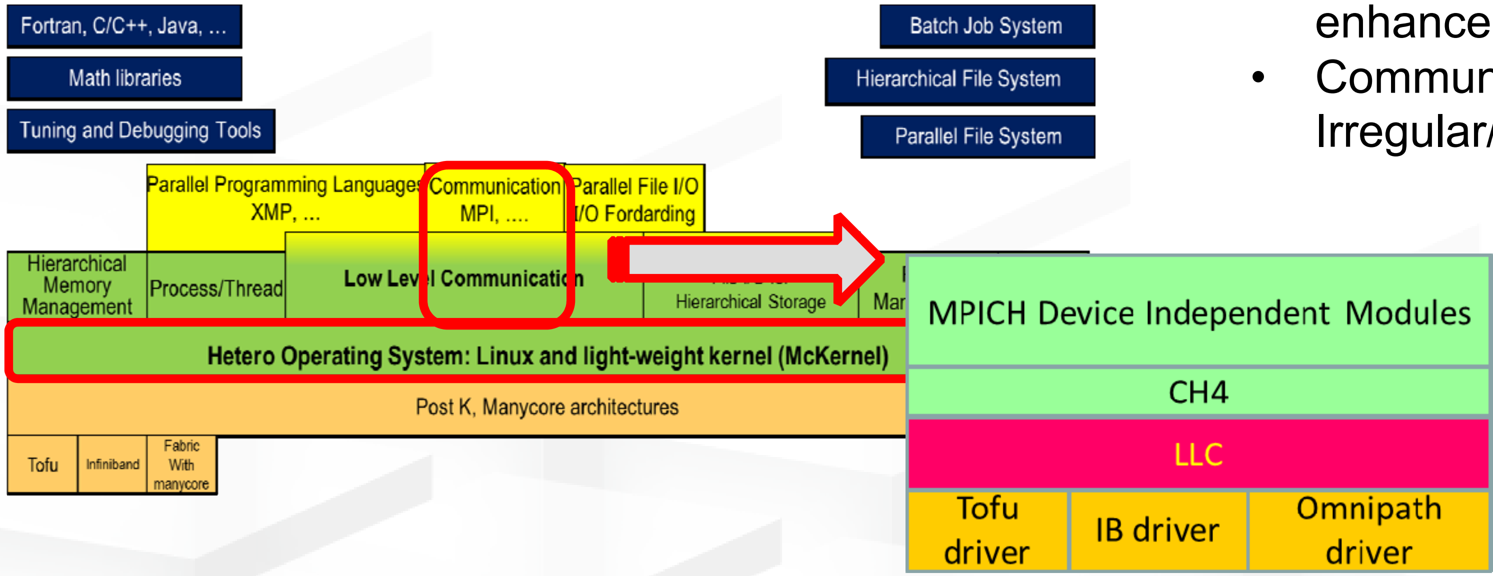
	Program	Brief description
①	GENESIS	MD for proteins
②	Genomon	Genome processing (Genome alignment)
③	GAMERA	Earthquake simulator (FEM in unstructured & structured grid)
④	NICAM+LETK	Weather prediction system using Big data (structured grid stencil & ensemble Kalman filter)
⑤	NTChem	molecular electronic (structure calculation)
⑥	FFB	Large Eddy Simulation (unstructured grid)
⑦	RSDFT	an ab-initio program (density functional theory)
⑧	Adventure	Computational Mechanics System for Large Scale Analysis and Design (unstructured grid)
⑨	CCS-QCD	Lattice QCD simulation (structured grid Monte Carlo)

Co-design Elements in System Software

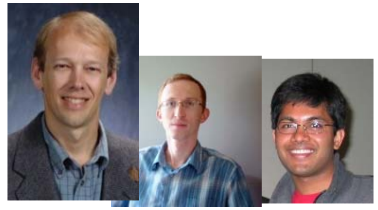
		Co-design Item
OS Kernel	*Applications*	Scheduler, Memory management
		In-situ Workloads (Visualization)
		Supporting efficient consecutive job execution
		PGAS model
	Node Architecture	Special memory allocation for NUMA domain process/thread
		Efficient intra-node MPI execution (process vs. thread)
Communication	*Applications*	Collective operations, nonblocking operations, scalability
	Node Architecture	Optimization for many NUMA domains
		Applicability of RDMA-based Communication
I/O	*Applications*	netCDF API extension for application domains to reduce pressure on the file system
		Data exchange between applications (Coupling)
		Location of temporal files based on workflows and memory availability (possibly in RAM)
	Storage Architecture	Async. I/O, Caching/Buffering to reduce pressure on I/O network and the file system
		Methods for massive files

International Collaboration

		Co-design Item
OS Kernel	*Applications*	Scheduler, Memory management
		In-situ Workloads (Visualization)
		Supporting efficient consecutive job execution
		PGAS model
	Node Architecture	Special memory allocation for NUMA domain process/thread
		Efficient intra-node MPI execution (process vs. thread)
Communication	*Applications*	Collective operations, nonblocking operations, scalability
	Node Architecture	Optimization for many NUMA domains
		Applicability of RDMA-based Communication



Argonne NATIONAL LABORATORY



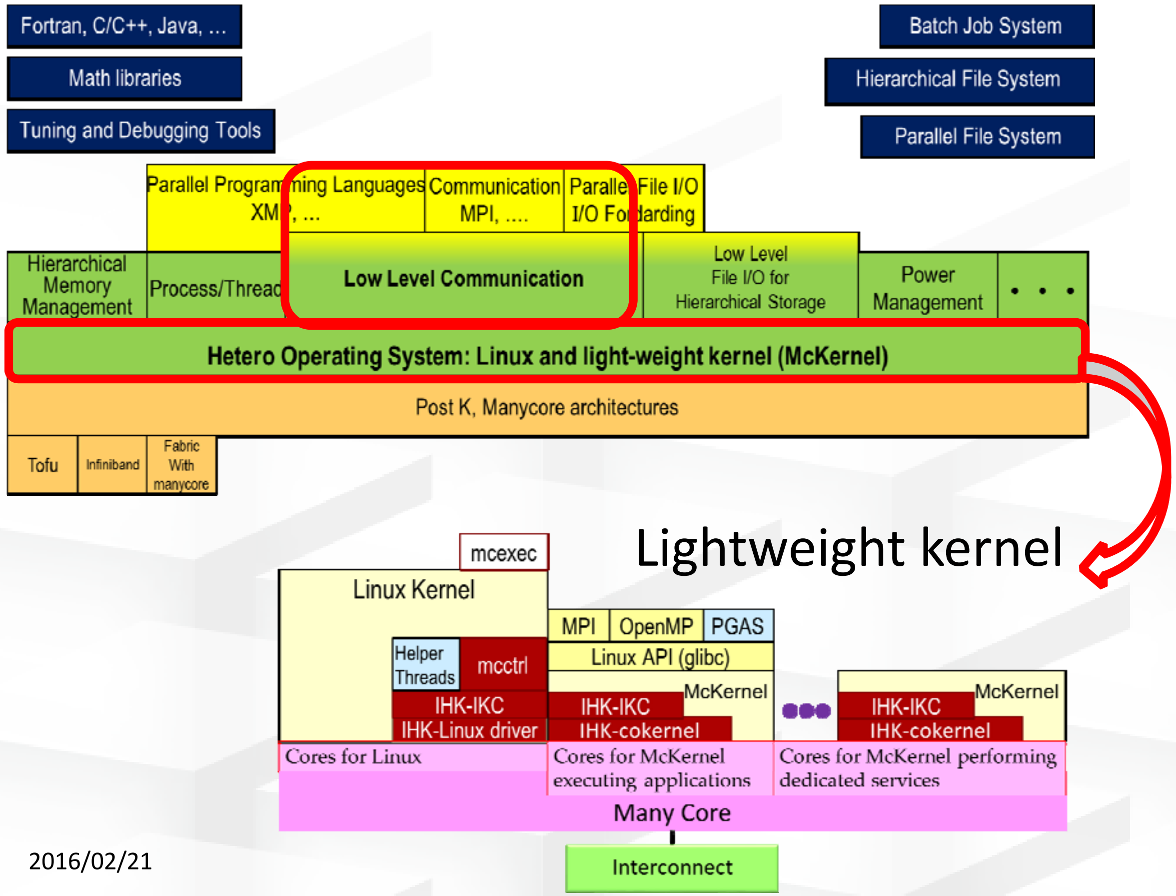
- Memory management for new memory hierarchy
- MPICH and LLC communication libraries
  - Scalability and performance enhancements to communication library
  - Communication Enhancements for Irregular/Dynamic Environments Pavan

In terms of advanced software development

- Argonne contribution: CH4 hackathon for LLC
- AICS contribution: a part of CH4 implementation

International Collaboration

		Co-design Item
OS Kernel	*Applications*	Scheduler, Memory management
		In-situ Workloads (Visualization)
		Supporting efficient consecutive job execution
		PGAS model
	Node Architecture	Special memory allocation for NUMA domain process/thread
		Efficient intra-node MPI execution (process vs. thread)
Communication	*Applications*	Collective operations, nonblocking operations, scalability
	Node Architecture	Optimization for many NUMA domains
		Applicability of RDMA-based Communication



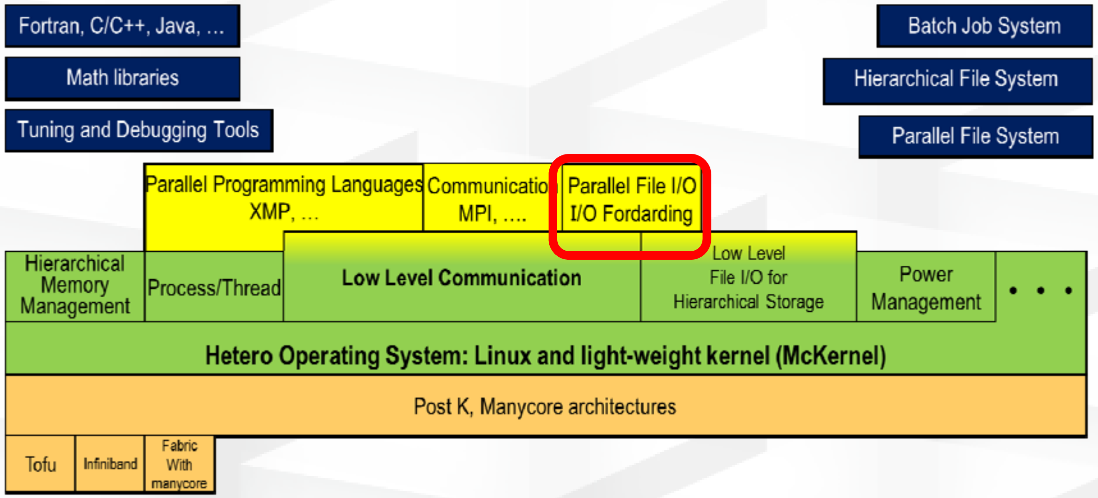
intel

- Understanding benefit of lightweight kernel
- Understanding differences of McKernel and mOS
- Standardization of API for lightweight kernel (Plan)

- Twice meetings per year
- A researcher visits Intel for a few months

International Collaboration

		Co-design Item
OS Kernel	*Applications*	Scheduler, Memory management
		In-situ Workloads (Visualization)
		Supporting efficient consecutive job execution
		PGAS model
	Node Architecture	Special memory allocation for NUMA domain process/thread
		Efficient intra-node MPI execution (process vs. thread)
Communication	*Applications*	Collective operations, nonblocking operations, scalability
	Node Architecture	Optimization for many NUMA domains
		Applicability of RDMA-based Communication
I/O	*Applications*	netCDF API extension for application domains to reduce pressure on the file system
		Data exchange between applications (Coupling)
		Location of temporal files based on workflows and memory availability (possibly in RAM)
	Storage Architecture	Async. I/O, Caching/Buffering to reduce pressure on I/O network and the file system
		Methods for massive files



Northwestern University

I/O Benchmarks and netCDF implementations for Scientific Big Data

A target application and its I/O problem will be shown in the following slides

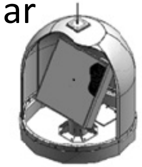
File I/O for Big data

PI: Takemasa Miyoshi, RIKEN AICS
"Innovating Big Data Assimilation technology for revolutionizing very-short-range severe weather prediction"

An innovative 30-second super-rapid update numerical weather prediction system for 30-minute/1-hour severe weather forecasting will be developed, aiding disaster prevention and mitigation, as well as bringing a scientific breakthrough in meteorology.

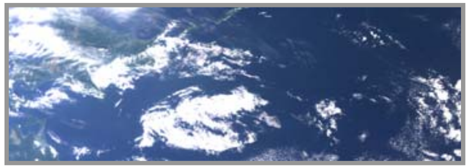
Phased array weather radar

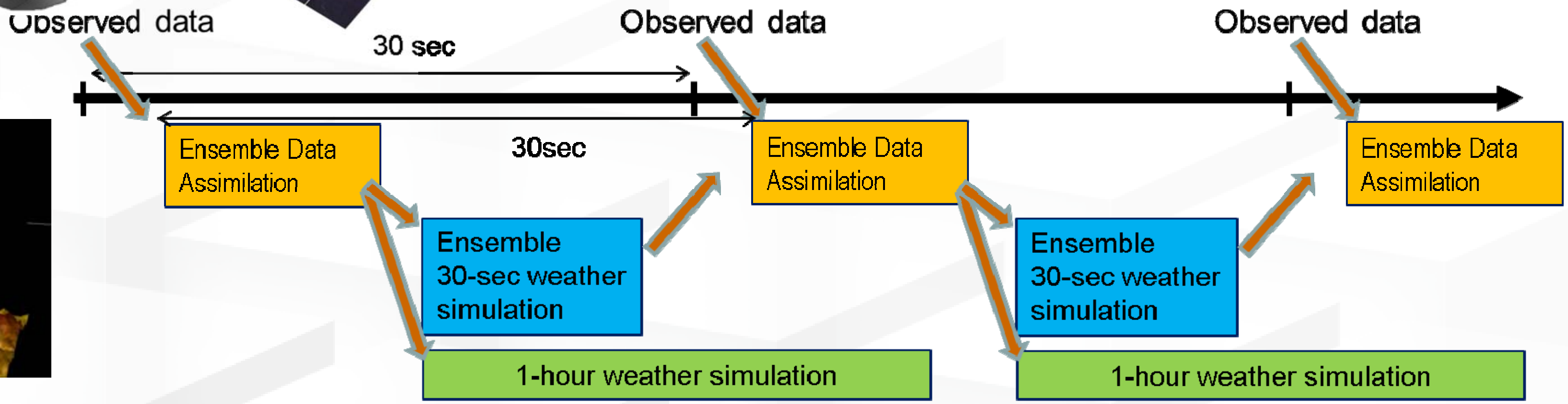
151.2 MB/30sec

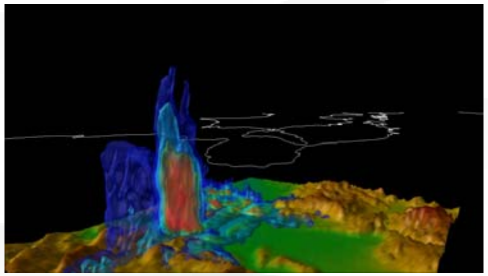


Satellite

500 MB/2.5min



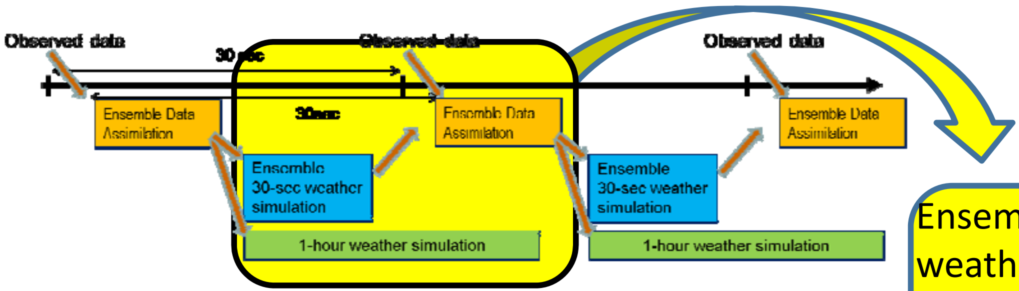


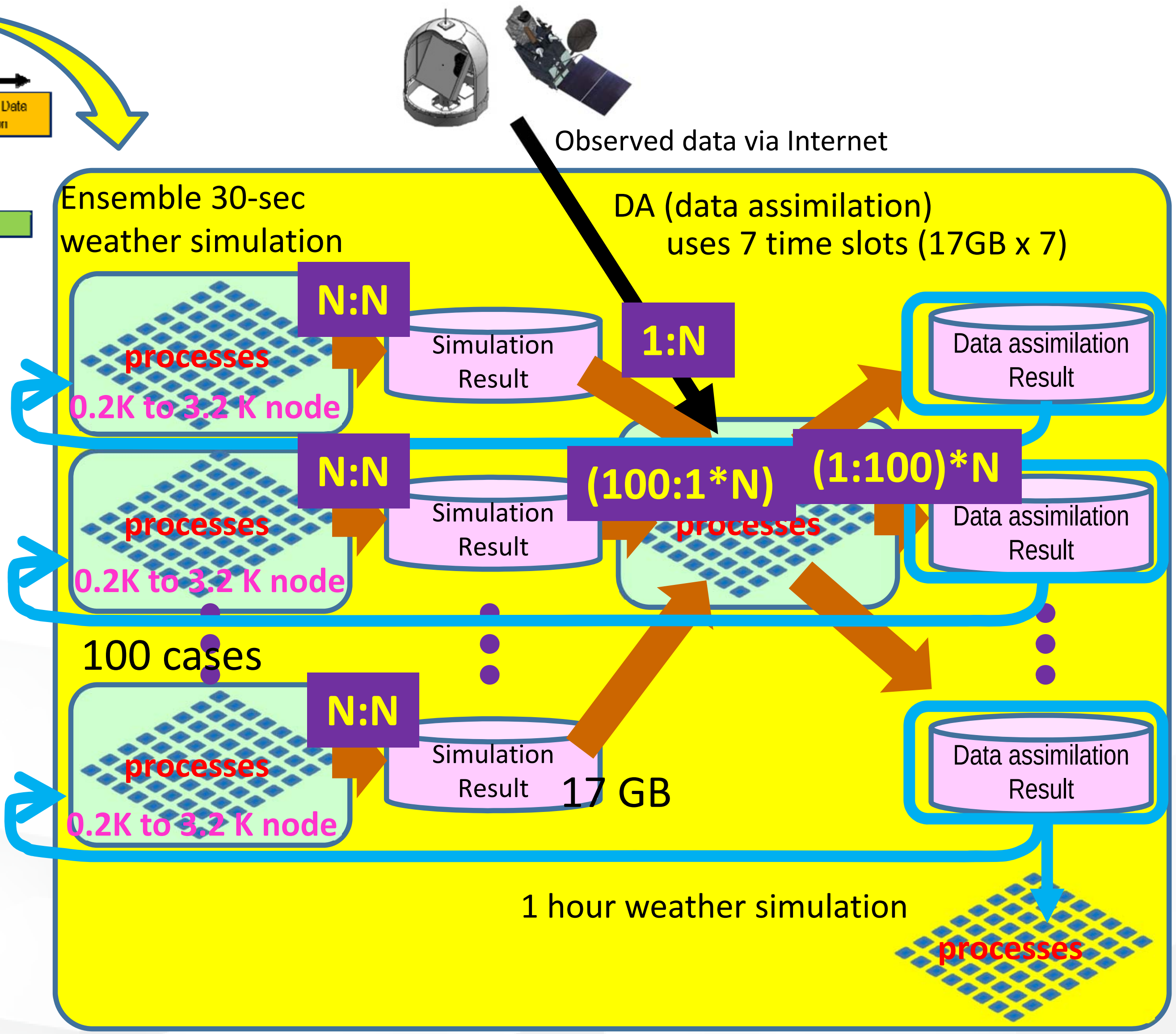


Rain particle

To meet real-timeness, 30 second responsibility, data exchange between data assimilation and weather simulations must be fast as much as possible

File I/O patterns in Ensemble simulation and data assimilation



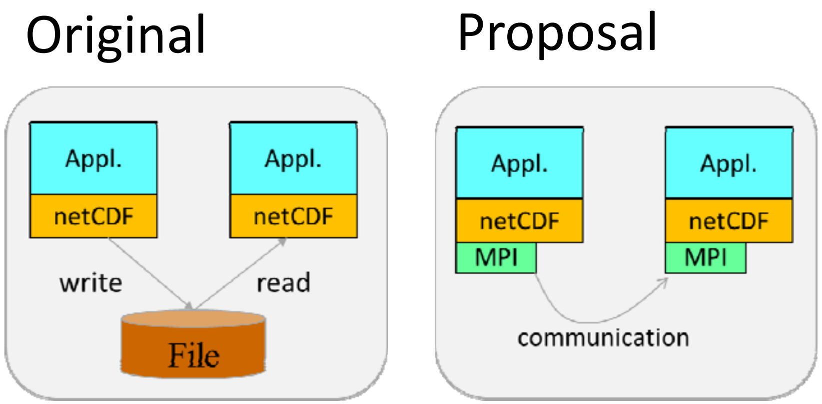


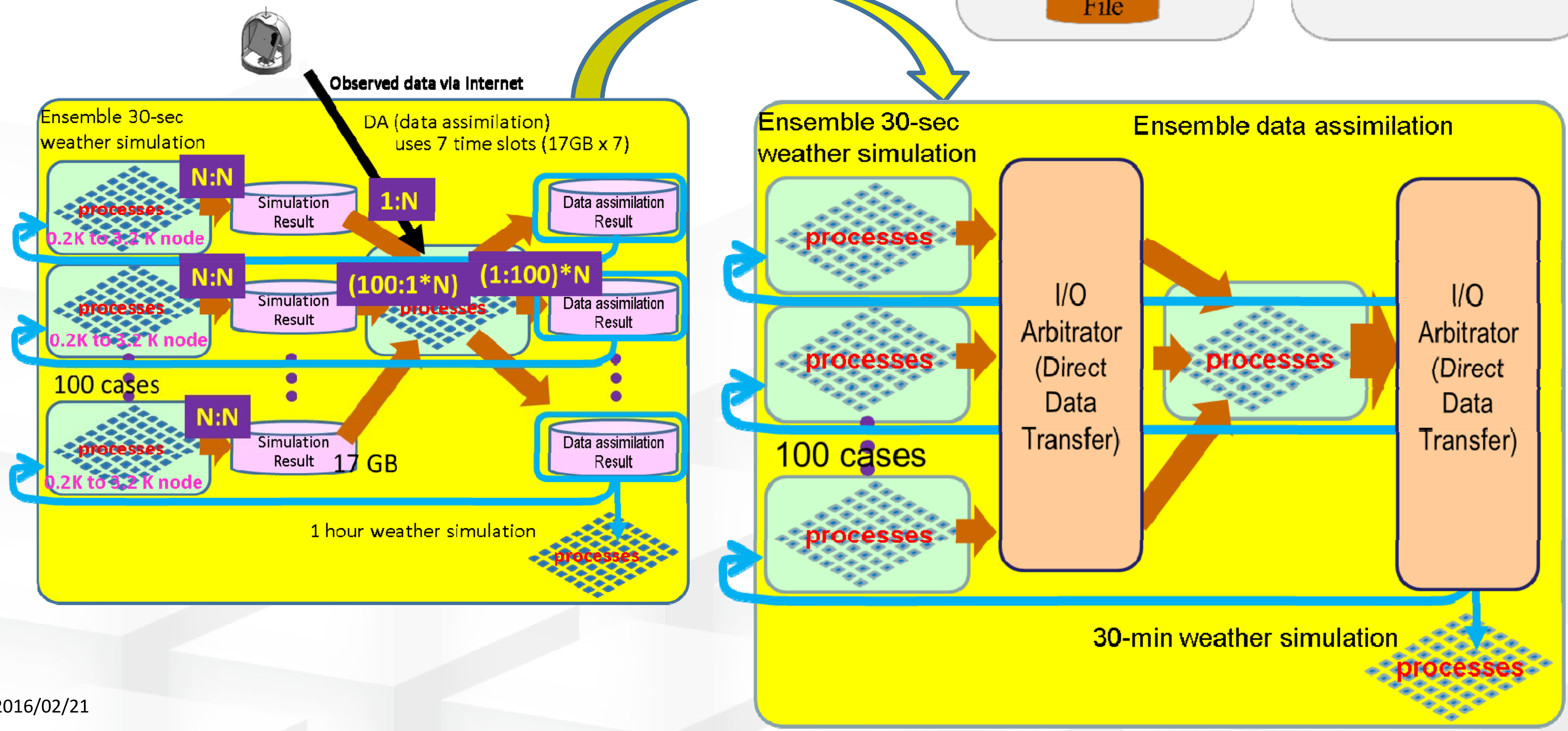
Every 30 second

1. Each weather simulation generates 17 GB data. 1.7 TB in total for 100 cases
2. Data assimilation code reads all results (17GB) and observed data (--1GB), and outputs assimilated data for 100 cases. That is, 1.7 TB in total for 100cases

Approach: I/O Arbitrator

- **Keeping the netCDF file I/O API**
- **Introducing additional API in order to realize direct data transfer without storing data into storage**
  - E.g., asynchronous I/O

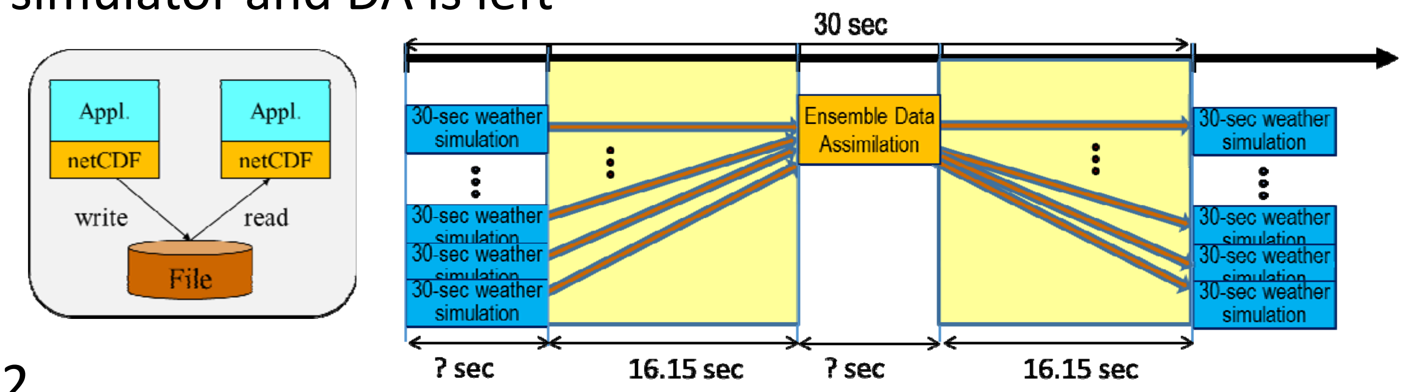




Prototype System Evaluation at RIKEN AICS

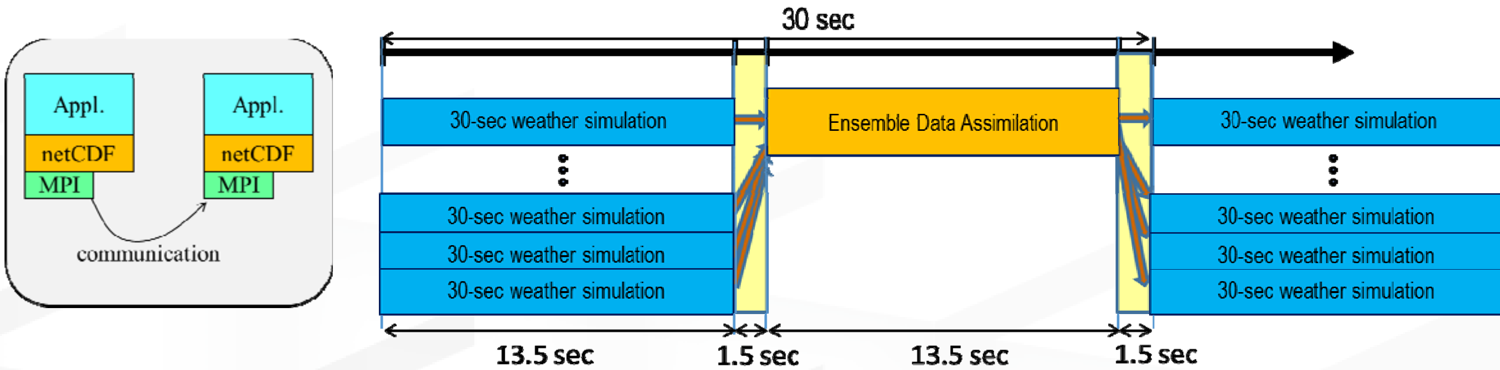
- File I/O: 16.15 sec x 2

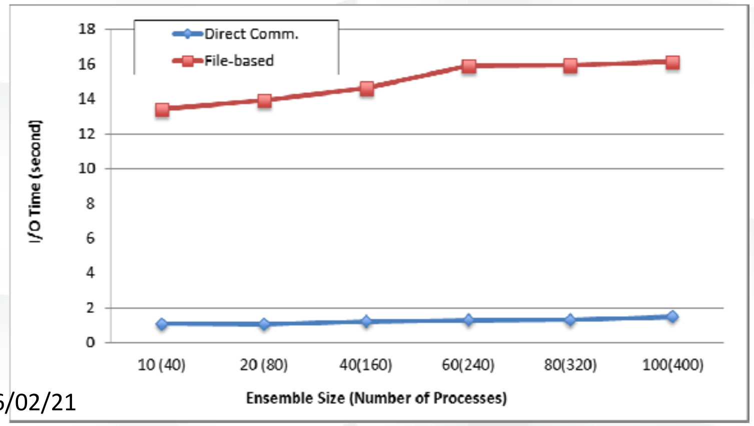
  No execution time for simulator and DA is left



- netCDF/MPI: 1.5 sec x 2

  Simulator and DA have 13.5 sec execution time





International Collaboration with Northwestern Univ.

- **Develop parallel I/O modules using PnetCDF**
  - Parallel netCDF is a parallel I/O library for accessing files in netCDF formats
  - Using the shared-file strategy that produces only one history file and one restart file
  - Restructure source codes to first define all variables and then write them
    - Avoids repeatedly entering and exiting define mode (expensive)
  - Use PnetCDF nonblocking APIs to aggregate multiple small requests into large ones for better performance

Concluding Remarks in Post K development

- **The basic architecture design has been reviewed by the MEXT evaluation committee**
- **The design and implementation phase starts**

- **The system software stack for Post K is being designed and implemented with the leverage of international collaborations**