Challenges on Programming Models and Languages for Post-Petascale Computing

-- from Japanese NGS project "The K computer" to Exascale computing --

Mitsuhisa Sato

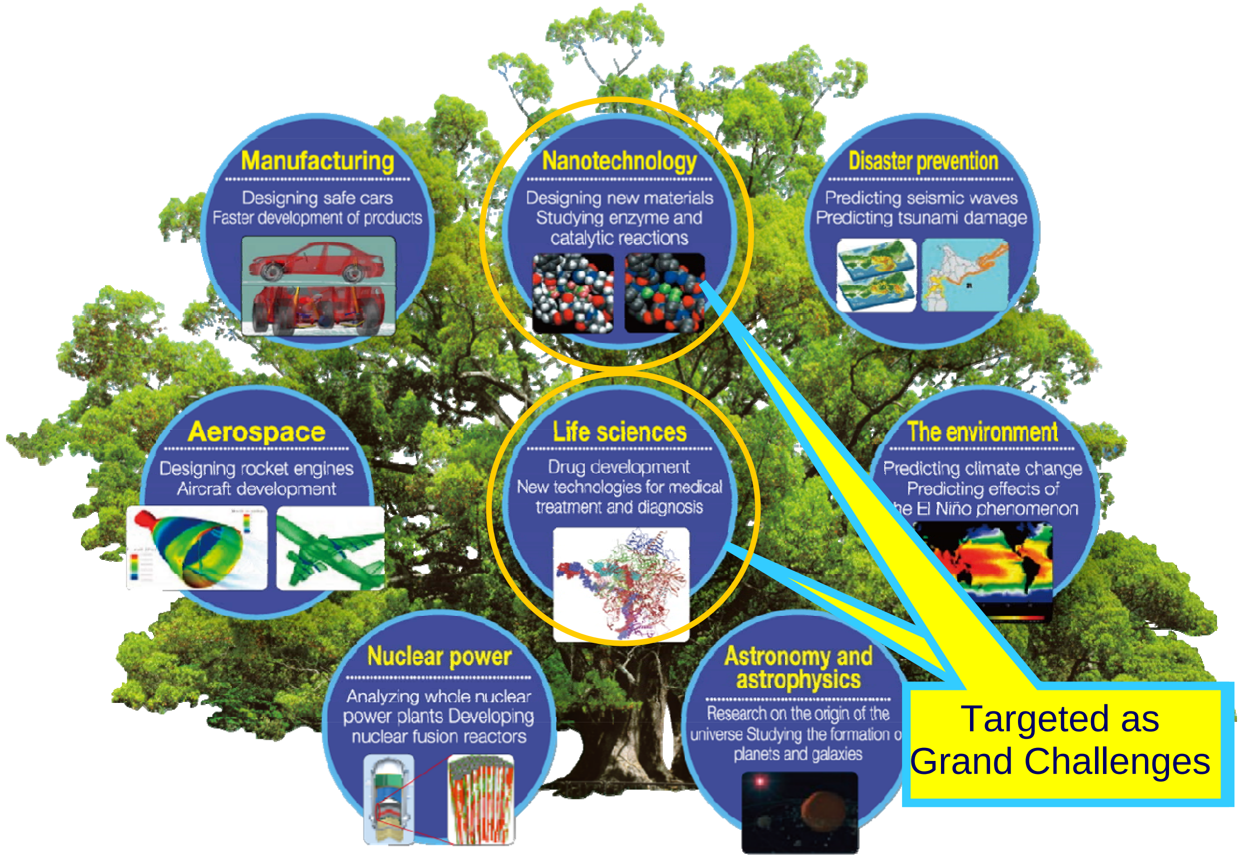
Center for Computational Sciences (CCS), University of Tsukuba & Advanced Institute for Computational Science (AICS), RIKEN

Agenda

- Update of Japanese NGS project
  - Objectives and goals, schedule …
  - "The K computer"
  - Organizations and Projects to NGS and beyond…
- XcalableMP: Programming model and language for Petascale computing
  - Status and performance
- Issues and projects for "post-petascale" computing …
  - Japanese-France FP3C (Framework Programming for Post Petascale Computing) project

Goals of the NGS (next generation supercomputer) project

- Development and installation of the most advanced high performance supercomputer system with LINPACK performance of 10 petaflops.
- Development and deployment of application software, which should be made to attain the system maximum capability, in various science and engineering fields.
- Establishment of an "Advanced Computational Science and Technology Center (tentative)" -> AICS as one of the Center of Excellences around supercomputing facilities.


Manufacturing
Designing safe cars
Faster development of products
Nanotechnology
Designing new materials
Studying enzyme and catalytic reactions
Disaster prevention
Predicting seismic waves
Predicting tsunami damage
Aerospace
Designing rocket engines
Aircraft development
Life sciences
Drug development
New technologies for medical treatment and diagnosis
The environment
Predicting climate change
Predicting effects of the El Niño phenomenon
Nuclear power
Analyzing whole nuclear power plants Developing nuclear fusion reactors
Astronomy and astrophysics
Research on the origin of the universe Studying the formation of planets and galaxies
Targeted as Grand Challenges

Schedule of the project

We are here.

		FY2006	FY2007	FY2008	FY2009	FY2010	FY2011	FY2012
System		Conceptual design		Detailed design	Prototype, evaluation	Production, installation, and adjustment		Tuning and improvement
Applications	Next-Generation Integrated Nanoscience Simulation			Development, production, and evaluation			Verification	
	Next-Generation Integrated Life Simulation			Development, production, and evaluation				Verification
Buildings	Computer building		Design		Construction			
	Research building		Design		Construction			

Open to the project

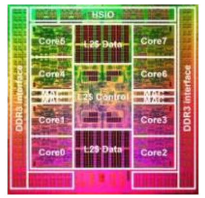
The K Supercomputer

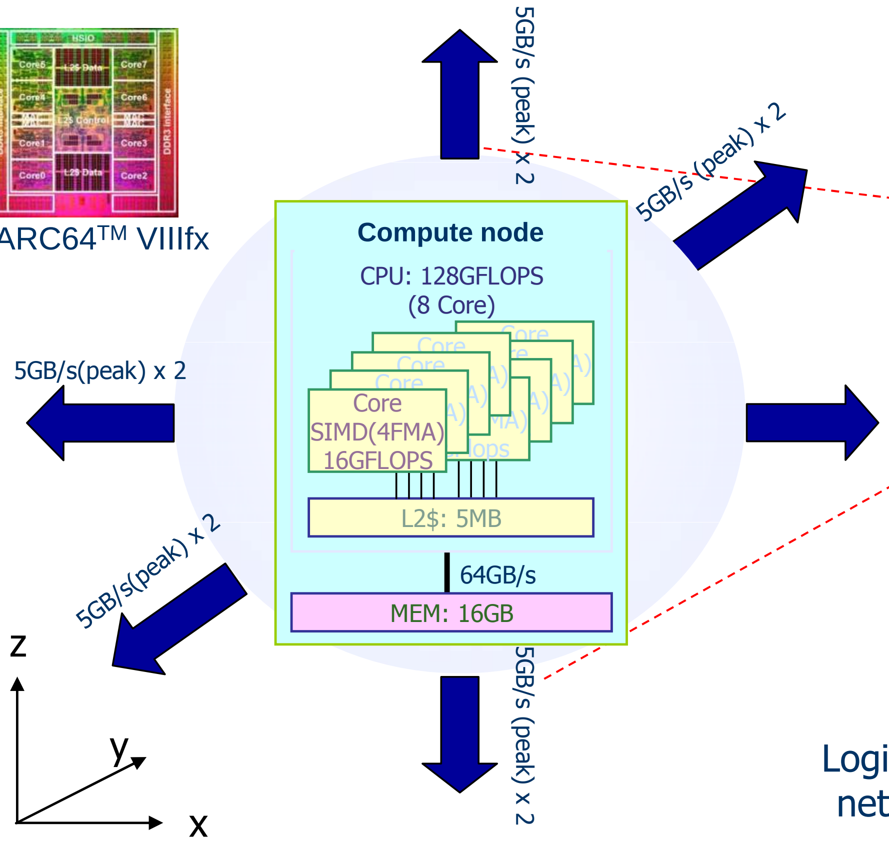
System configuration
and software

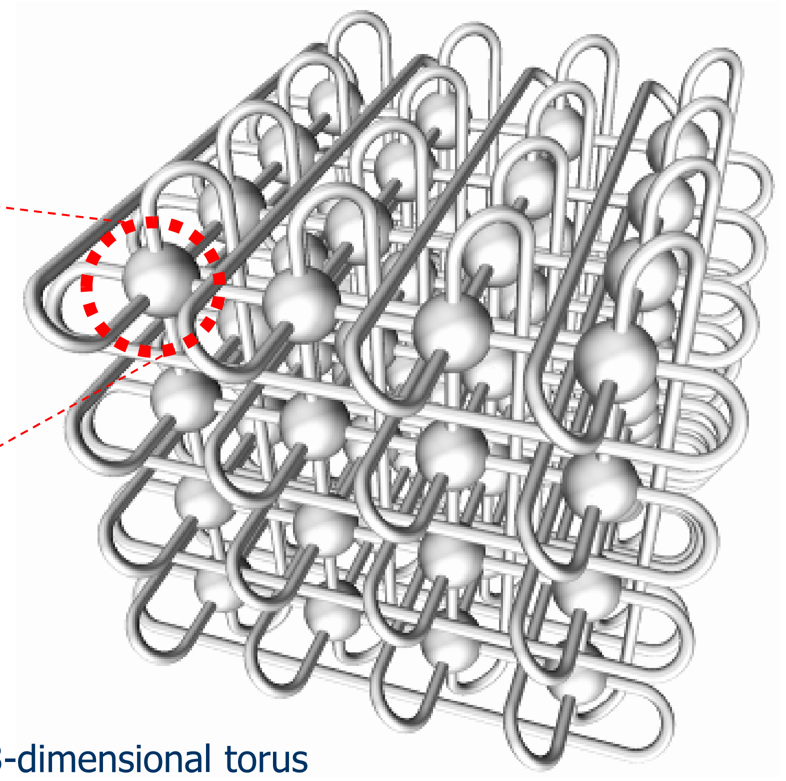
Compute Nodes and network

- Compute nodes (CPUs): > 80,000
  - Number of cores: > 640,000
- Peak performance: > 10PFLOPS
- Memory: > 1PB (16GB/node)

- Logical 3-dimensional torus network
- Peak bandwidth: 5GB/s x 2 for each direction of logical 3-dimensional torus network
- bi-section bandwidth: > 30TB/s


SPARC64TM VIIIfx



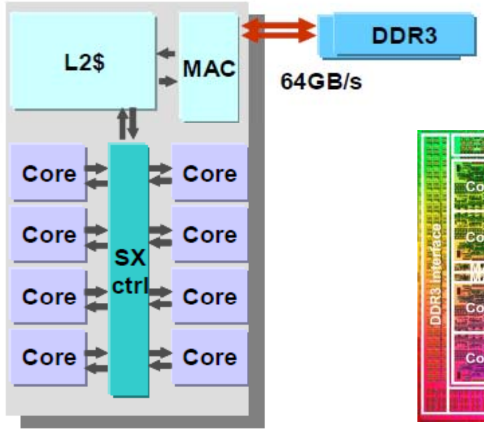

Logical 3-dimensional torus network for programming

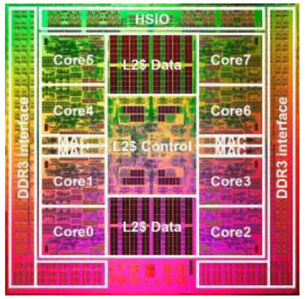
Courtesy of FUJITSU Ltd.

CPU Features (Fujitsu SPARC64^{TM} VIIIfx)

- 8 cores
- 2 SIMD operation circuit
  - 2 Multiply & add floating-point operations (SP or DP) are executed in one SIMD instruction
- 256 FP registers (double precision)
- Shared 5MB L2 Cache (10way)
  - Hardware barrier
  - Prefetch instruction
  - Software controllable cache
    - Sectored cache
- Performance
  - 16GFLOPS/core, 128GFLOPS/CPU

16GF/core(2*4*2G)



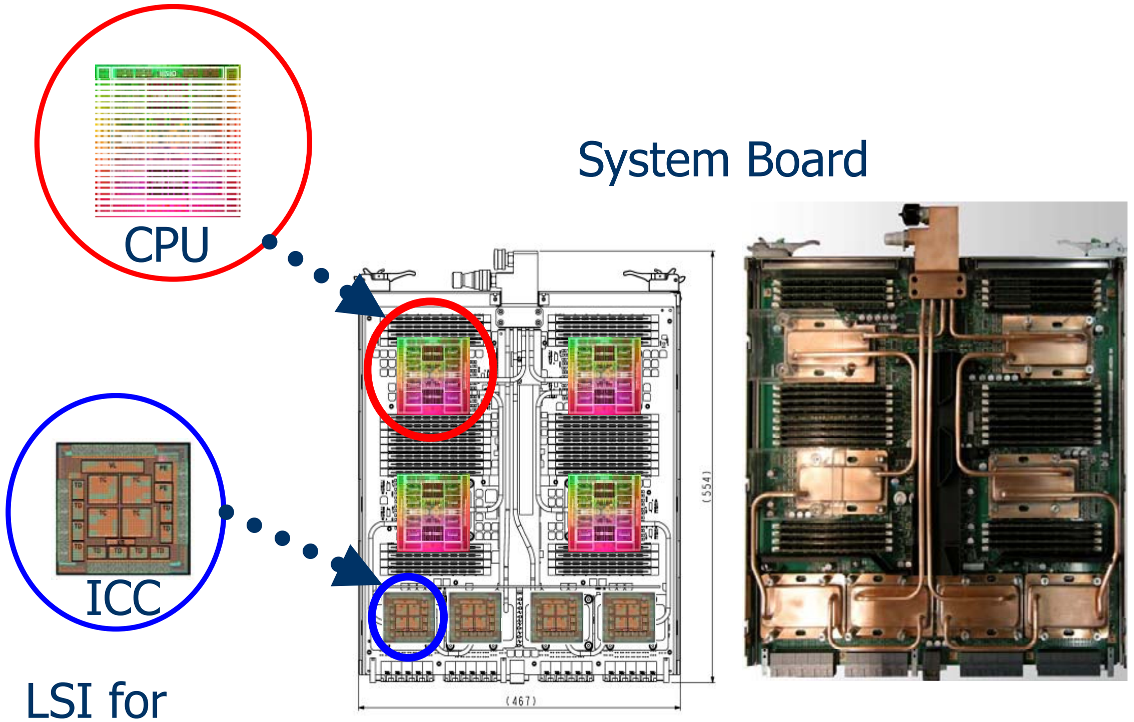


45nm CMOS process, 2GHz
22.7mm x 22.6mm
760 M transisters
58W（at 30℃ by water cooling)

Reference: SPARC64^{TM} VIIIfx Extensions
http://img.jp.fujitsu.com/downloads/jp/jhpc/sparc64viiifx-extensions.pdf

Photo of proto-type system

- Prototype system has been built.
  - Several system boards are compiled and set into a cabinets.



LSI for interconnect

Courtesy of FUJITSU Ltd.

Organizations and Projects to NGS and beyond …

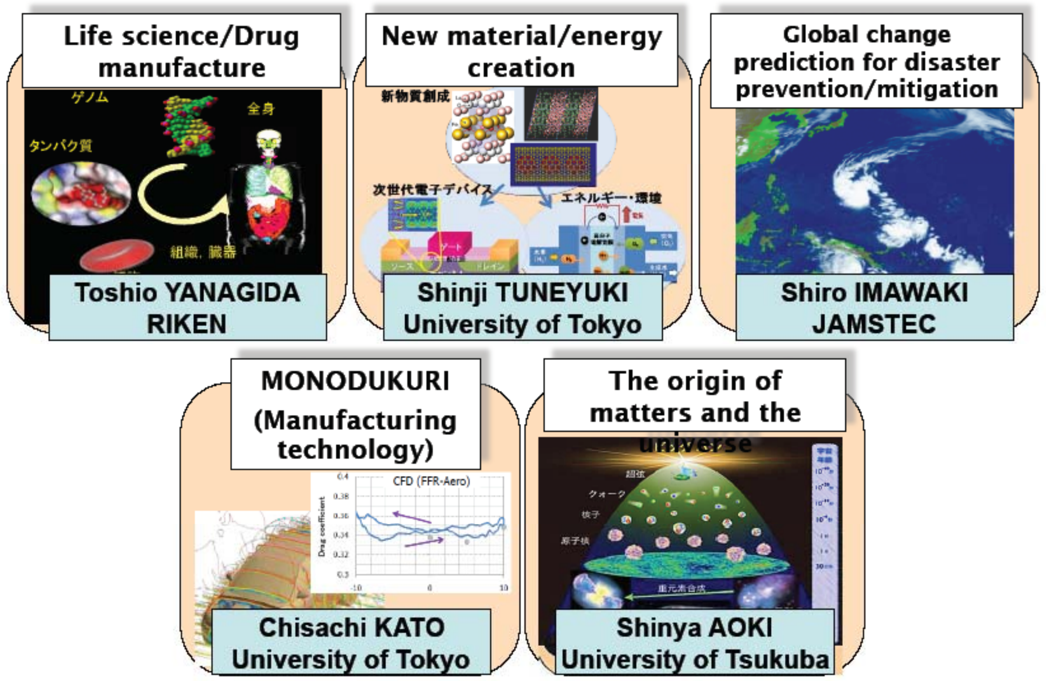
- Funds for Core organizations for 5 strategic fields
- Consortium and High-performance Computing Infrastructure (HPCI)
- RIKEN Advanced Institute for Computational Science (AICS)

How to organize users of NGS

- Strategic Use: MEXT selected 5 strategic fields from national point of view
    - Field 1: Life science/Drug manufacture
    - Field 2: New material/energy creation
    - Field 3: Global change prediction for disaster prevention/mitigation
    - Field 4: *Mono-zukuri* (Manufacturing technology)
    - Field 5: The origin of matters and the universe (core org. CCS, U. Tsukuba)
  - MEXT funds 5 core organizations that lead research activities in these 5 strategic fields

- General Use:

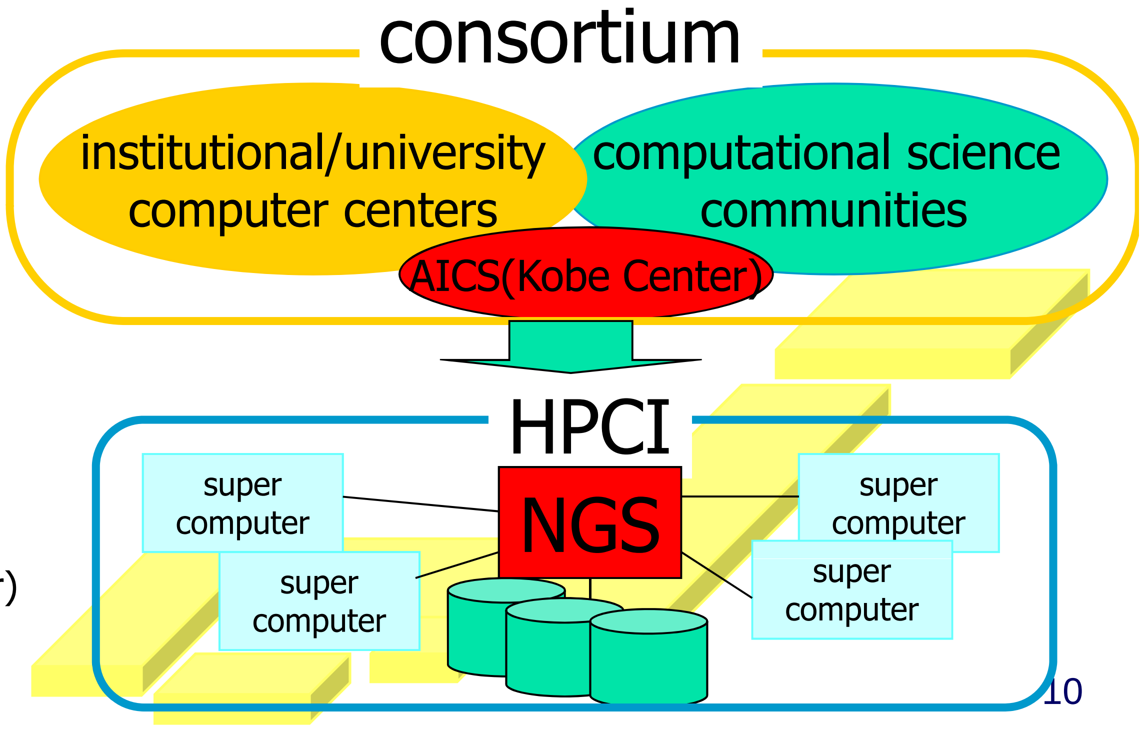
  For the needs of the researchers in many science and technology fields including industrial use and educational use



Life science/Drug manufacture
ゲノム
全身
タンパク質
組織、臓器
Toshio YANAGIDA RIKEN
New material/energy creation
新物質創成
次世代電子デバイス
エネルギー・環境
Shinji TUNEYUKI University of Tokyo
Global change prediction for disaster prevention/mitigation
Shiro IMAWAKI JAMSTEC
MONODUKURI (Manufacturing technology)
CFD (FFR-Aero)
Drag coefficient
Chisachi KATO University of Tokyo
The origin of matters and the universe
クォーク
原子核
Shinya AOKI University of Tsukuba

Consortium and High-performance Computing Infrastructure (HPCI)

- Background:
  - The goal of the NGS has been reconsidered by the new government for accountability for "taxpayers" : "Creation of the Innovative High-Performance Computing Infrastructure (HPCI)".
- HPCI: High-Performance Computing Infrastructure
  - Enable Integrated operation of NGS with other institutional supercomputers
  - Provide seamless access from supercomputers and user's machines to NGS.
  - Provide large-scale storage systems shared by NGS and others.
- HPCI (or HPC) Consortium
  - To play a role as a main body to run HPCI (and design HPCI).
  - To organize computational science communities from several application fields and institutional/university supercomputer centers.
    - Including AICS (Kobe Center)



RIKEN Advanced Institute for Computational Science (AICS)

- The institute have been established at the NGS in Kobe (started in October 2010)
- Objectives:
  - Take responsibility to run the NGS
  - Carry out the leading edge of computational science technologies and contribute for COE of computational science in Japan
  - Propose the future directions of HPC in Japan and conduct it.
- Topics
  - Promoting strong collaborations between computational and computer scientists, working with core-organizations of each fields together.
  - Fostering young scientists who exploit both computational and computer science
  - Research for new concepts for HPC in the future beyond the NGS (this is, exascale?)

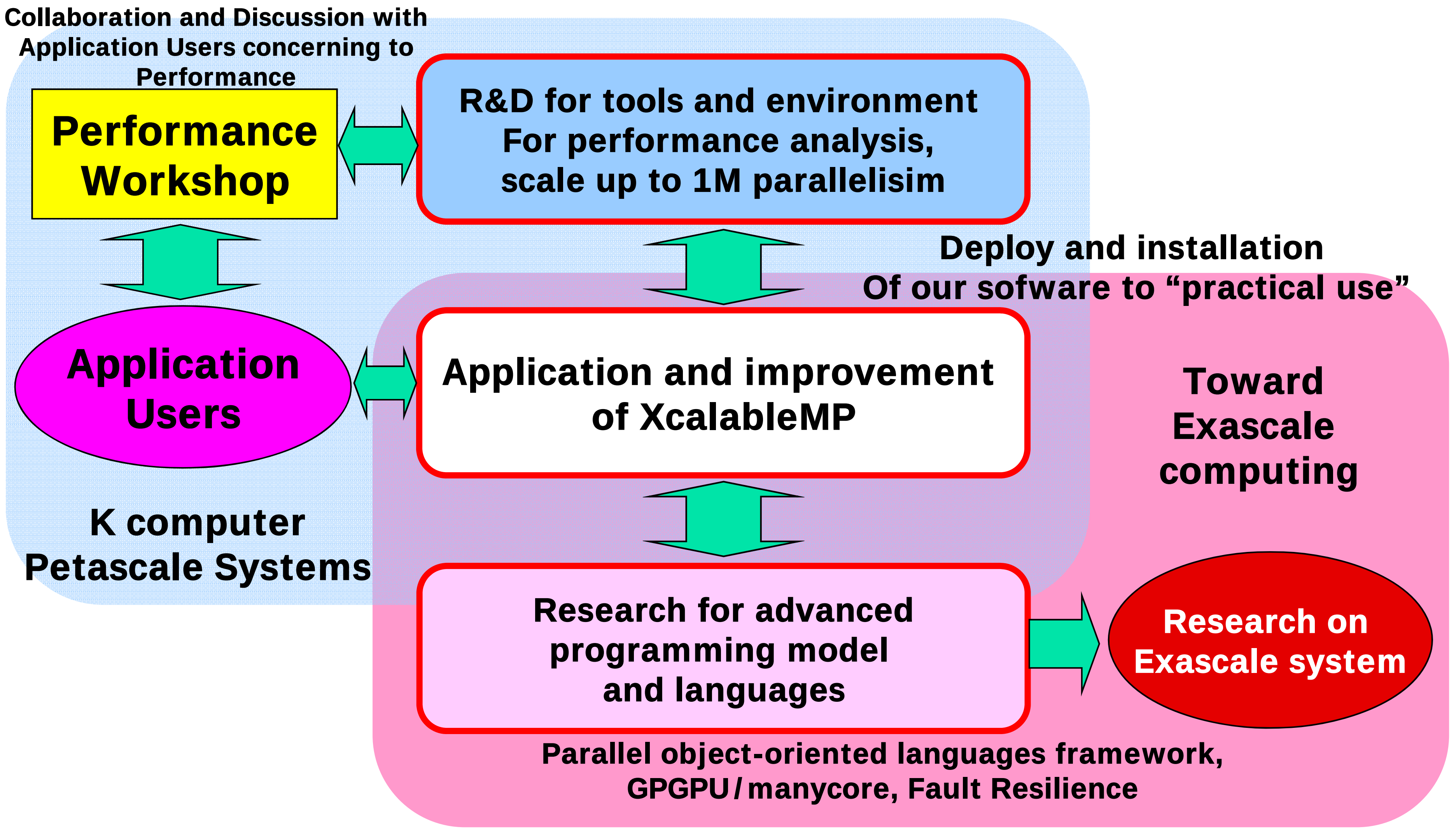
http://www.aics.riken.jp/

Research teams in AICS

- Computer science
  - System software (leader: Yutaka Ishikawa)
  - Programming environment (leader: Mitsuhisa Sato)
  - Processor/accelerator (leader: Makoto Taiji)
- Computational Science (note: team names are tentative)
  - Particle Physics (leader: Yoshinobu Kuramashi)
  - Climate (leader: Hirofiumi Tomita)
  - Condensed Matter Physics (leader: Seiji Yunoki)
  - Chemistry(leader: Takahito Nakajima)
  - Biochemistry (leader: Yuji Sugita)

Research Agenda in "Programming Environment" team in AICS

The technology of programming models/languages and environment plays an important role to bridge between programmers and systems. Our team will perform researches for applicaitons to exploit full potentials of large-scale parallelism in our petascale system (K computer) by providing practical parallel programming languages and performance tools. And we also will explore programming technologies towards the next generation "exascale" computing.


Collaboration and Discussion with Application Users concerning to Performance
Performance Workshop
R&D for tools and environment For performance analysis, scale up to 1M parallelisim
Deploy and installation Of our sofware to "practical use"
Application Users
Application and improvement of XcalableMP
Toward Exascale computing
K computer Petascale Systems
Research for advanced programming model and languages
Research on Exascale system
Parallel object-oriented languages framework, GPGPU/manycore, Fault Resilience

What's XcalableMP?

- XcalableMP (XMP for short) is:
  - A programming model and language for distributed memory , proposed by XMP WG
  - http://www.xcalablemp.org

- XcalableMP Specification Working Group (XMP WG)
  - XMP WG is a special interest group, which organized to make a draft on "petascale" parallel language.
  - Started from December 2007, the meeting is held about once in every month.
    - Mainly active in Japan, but open for everybody.

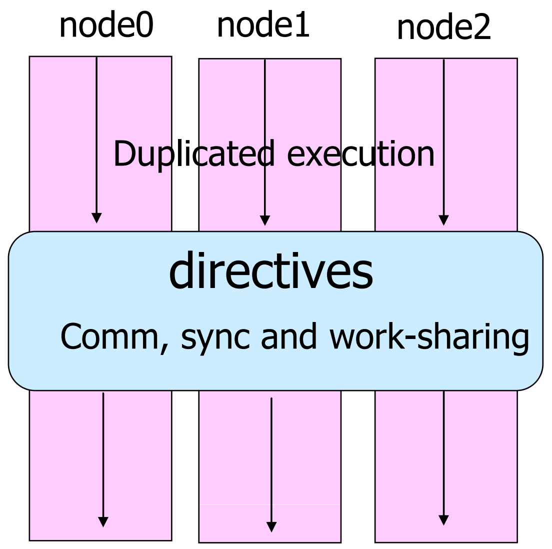
- XMP WG Members (the list of initial members)
  - Academia: M. Sato, T. Boku (compiler and system, U. Tsukuba), K. Nakajima (app. and programming, U. Tokyo), Nanri (system, Kyusyu U.), Okabe (HPF, Kyoto U.)
  - Research Lab.: Watanabe and Yokokawa (RIKEN), Sakagami (app. and HPF, NIFS), Matsuo (app., JAXA), Uehara (app., JAMSTEC/ES)
  - Industries: Iwashita and Hotta (HPF and XPFortran, Fujitsu), Murai and Seo (HPF, NEC), Anzaki and Negishi (Hitachi), (many HPF developers!)

- Funding for development
  - e-science project : "Seamless and Highly-productive Parallel Programming Environment for High-performance computing" project funded by MEXT,Japan
    - Project PI: Yutaka Ishiakwa, co-PI: Sato and Nakashima(Kyoto), PO: Prof. Oyanagi
    - Project Period: 2008/Oct to 2012/Mar (3.5 years)

XcalableMP(XMP) http://www.xcalablemp.org

- Programming model
  - Directive-based language extensions for Fortran and C for PGAS model
  - Global view programming with global-view distributed data structures for data parallelism
    - A set of threads are started as a logical task. Work mapping constructs are used to map works and iteration with affinity to data explicitly.
    - Rich communication and sync directives such as "gmove" and "shadow".
    - Many concepts are inherited from HPF
  - Co-array feature of CAF is adopted as a part of the language spec for local view programming (also defined in C).

- Execution model
  - SPMD as a basic execution model
    - A thread starts execution in each node independently (as in MPI) .
  - Duplicated execution if no directives are encountered
  - Node sets concept for Task parallelism



XcalableMP(XMP) http://www.xcalablemp.org

- Language status
  - XMP Spec Version 0.7 is available at XMP site.
  - XMP-IO and multicore extension are under discussion.
  - Prototype compilers and tools are being developed in the Japanese MEXT "e-science" project.

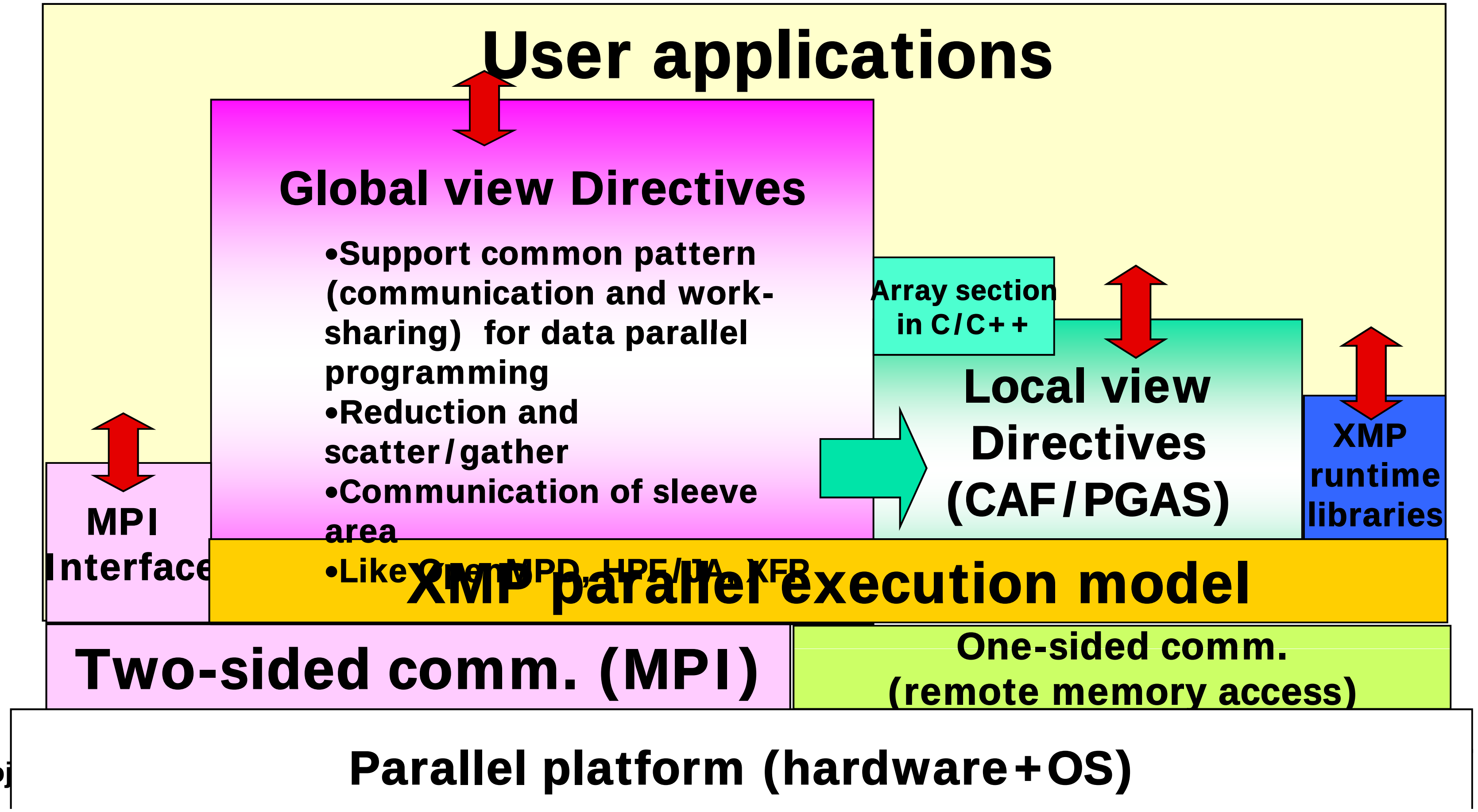
- Compiler & tools
  - XMP prototype compiler (xmpcc version 0.5) for C is available from U. of Tsukuba.
    - Open-source, C to C source compiler with the runtime using MPI
  - No specific tools available yet (MPI tools can be used).

- Platforms supported
  - Linux Cluster, Cray XT5 …
  - Any systems running MPI
    - The current runtime system designed on top of MPI

- Available Codes
  - Jacobi iterations
  - NBP kernels
  - HPC Challenge Benchmark

Overview of XcalableMP

- XMP supports typical parallelization based on the **data parallel paradigm** and work sharing under "**global view**“
  - An original sequential code can be parallelized with **directives**, like OpenMP.
- XMP also includes CAF-like PGAS (Partitioned Global Address Space) feature as "**local view**" programming.


User applications
Global view Directives
•Support common pattern (communication and work-sharing) for data parallel programming
•Reduction and scatter/gather
•Communication of sleeve area
•Like OpenMPD, HPF/JA, XFP
Array section in C/C++
Local view Directives (CAF/PGAS)
XMP runtime libraries
MPI Interface
XMP parallel execution model
Two-sided comm. (MPI)
One-sided comm. (remote memory access)
Parallel platform (hardware+OS)

Code Example

```
int array[YMAX][XMAX];

#pragma xmp nodes p(4)
#pragma xmp template t(YMAX)
#pragma xmp distribute t(block) on p
#pragma xmp align array[i][*] with t(i)

main(){
  int i, j, res;
  res = 0;

#pragma xmp loop on t(i)  reduction(+:res)
  for(i = 0; i < 10; i++)
    for(j = 0; j < 10; j++){
      array[i][j] = func(i, j);
      res += array[i][j];
    }
}
```

data distribution

add to the serial code : incremental parallelization

work sharing and data synchronization

Research Issues for Petascale computing

- Global view vs. Local view, one-sided comm. vs. two-sided comm.
  - PGAS langs such as UPC, CAF only support only local view and one-sided. Is it enough?
  - XMP global view programs are complied into two sided comm. It must be more efficient?
- Task parallelism
  - For multi-physics
- Multicore support
  - Parallel loops by “loop” directives can be extended to be executed in parallel between cores in socket.
  - Combination with OpenMP
- XMP-IO, IO integration
  - High performance parallel IO for distributed arrays.
  - MPI-IO based

Task concept in XcalableMP

- *Executing node set* : a set of nodes executing the same task
  - Collective operations (barrier ...) are done in the executing node set.
- A *task* is a specific instance of executable code and its data environments executed in a set of nodes.
- Task construct is used to execute a task.

```
!$xmp  task on node-set
        block
!$xmp  end task
```

```
if (the current node
belong to node-set)
then
  create executing
  node set
        block
endif
```

```
subroutine caller
!$xmp nodes p(1000)
       real a(100,100)
!$xmp tasks
!$xmp  task on p(1:500)
         call task1(a)
!$xmp  end task
!$xmp  task on p(501:800)
         call task1(a)
!$xmp  end task
!$xmp  task on p(801:1000)
         call task1(a)
!$xmp  end task
!$xmp end tasks
       end do
```

A task executing on p(1:500) is created, and execute subroutine "task1"

```
subroutine task1(a)
!$xmp nodes q(*) = *
 real a(100,100)
 …
 …
end subroutine
```

FP3C

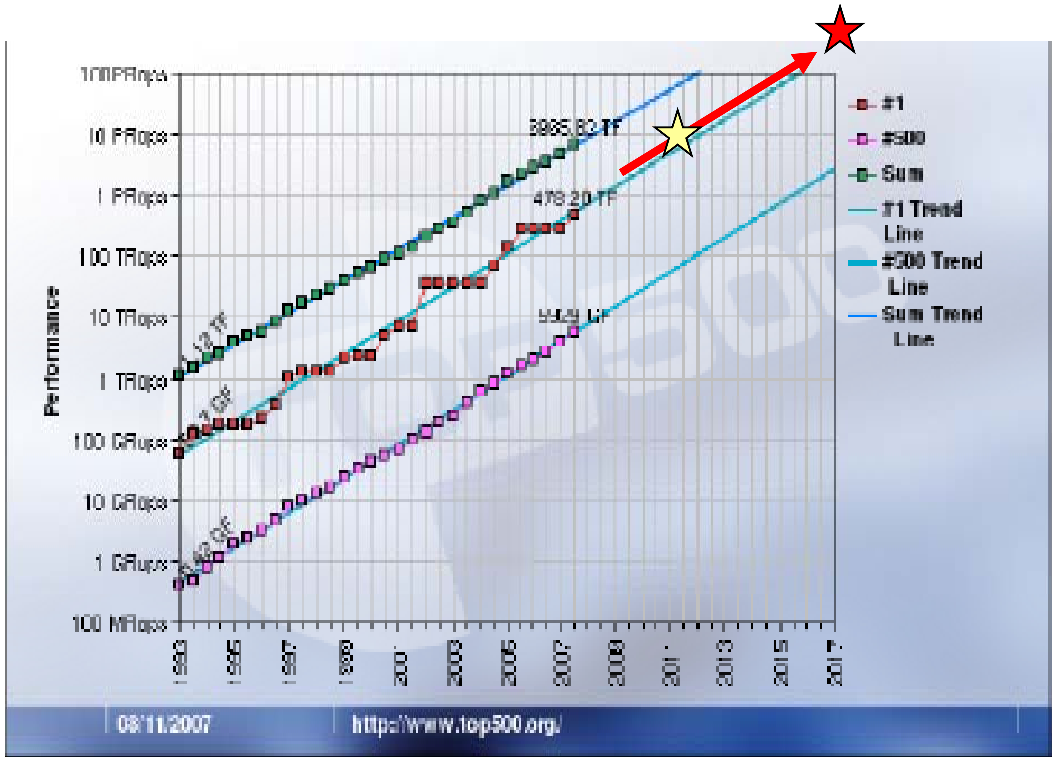
Issues for exascale computing …

- JST-ANR "Framework and Programming for Post Petascale Computing (FP3C)" project
  - France and Japan Fund: A collaborative call for proposals between "ANR-JST in ICT (Information and Communication Science and Technologies)" includes "Software and algorithm aspects of high performance computing (Axis 8)"
  - PI of Japan: M. Sato, PI of France: S. Petiton
  - 3 years from 2010 to 2012

Background: "Post-petascale computing“, toward exascale computing

FP3C

- State of the art: Petascale computing infrastructure
  - US: Blue Waters（10PF?, 2010-2011）,sequoia （20PF,201-2012）
  - Japan: NGS (>10PF, 2011-2012)
  - France/EU: PRACE machine (>1PF, 2012-2013?)
  - #cores 10^6
  - power >10 MW
- What's the next of "Petascale"?



What's "post-petascale" computing

FP3C

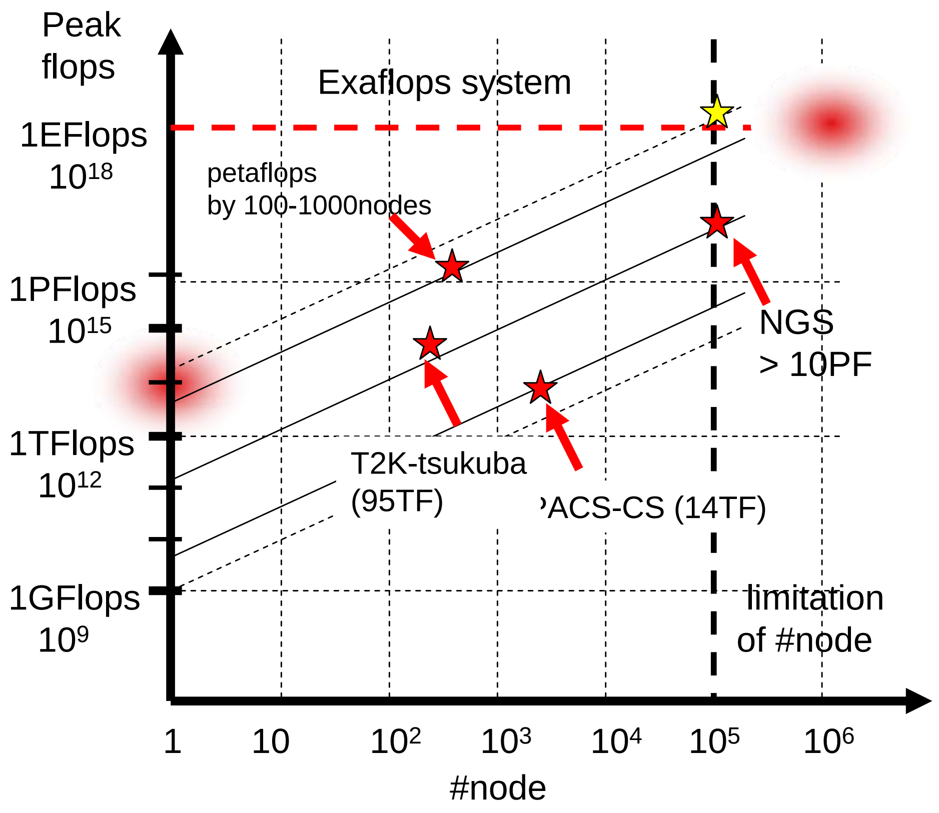
- Post-petascale ⊃ "exaflops"
  - Several Possibilities and Alternatives for the next of Petascale, on the road to "exascale"
- Exascale=Extreme Scale ≠ "exaflops"
  - Embedded terascale (hand-held, 10-100W)
  - Departmental petascale (1-2 racks, 10-100kW)
  - (Inter)national exascale (seveal 100 racks, 25-50MW)
- Challenges
  - strong scaling = find 1000× more parallelism in applications
  - fault tolerance = new algorithms + validation/verification
  - energy efficiency = new programming model(s), eg minimise data movement, intelligent powering
  - Novel hardware and programming, algorithms = GPGPUs, heterogeneous chips
  - massive (potentially corrupted) data and storage = new I/O models

Goal of our project

FP3C

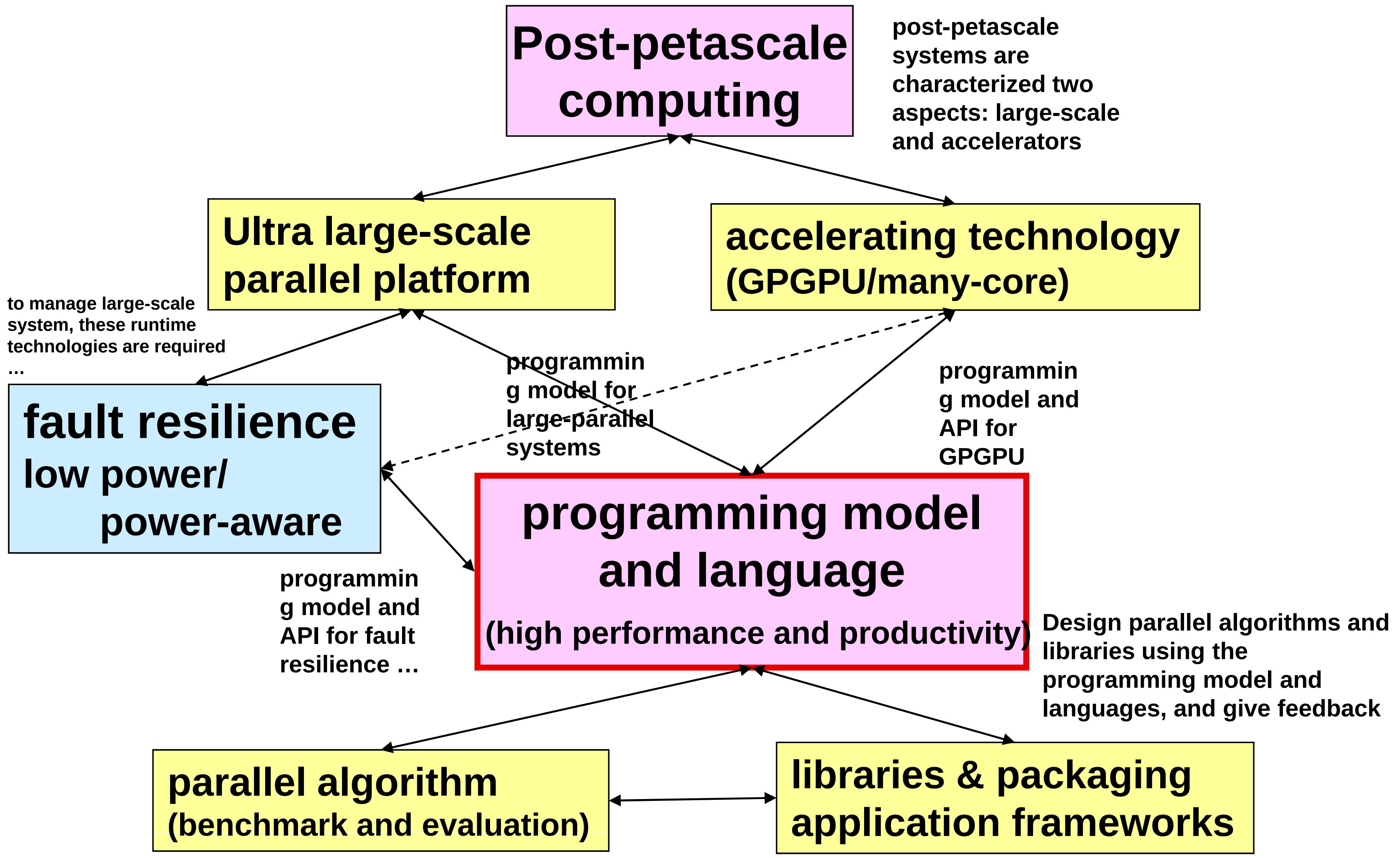
To contribute to establish software technologies, languages and programming models to explore extreme performance computing beyond petascale computing, on the road to exascale computing.

- Two important aspects of post-petascale computing
  - Ultra large-scale
    - > 10^6 nodes
  - Strong-scaling
    - > 10TFlops/node
    - accelerator, many-cores
- Software-driven approach
  - The main objective is to develop a programming chain and associated runtime systems which will allow scientific end-users to efficiently execute their applications on highly parallel and accelerated post-petascale platform.



Issues and our approaches on post-petascale computing

FP3C



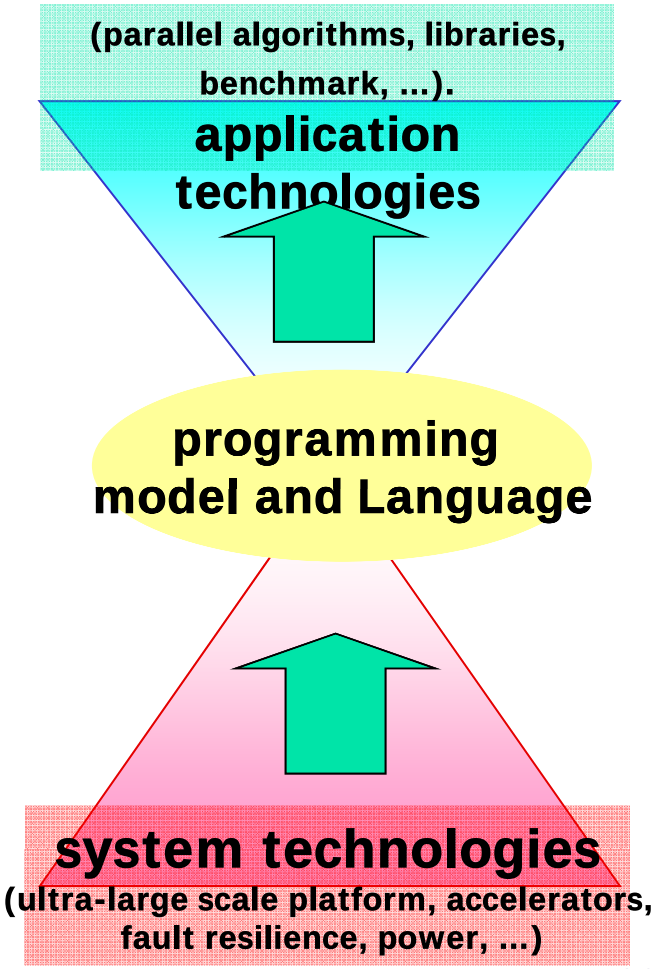
Post-petascale computing
post-petascale systems are characterized two aspects: large-scale and accelerators
Ultra large-scale parallel platform
accelerating technology (GPGPU/many-core)
to manage large-scale system, these runtime technologies are required …
fault resilience
low power/ power-aware
programmin g model for large-parallel systems
programmin g model and API for GPGPU
programming model and language
(high performance and productivity)
programmin g model and API for fault resilience …
Design parallel algorithms and libraries using the programming model and languages, and give feedback
parallel algorithm (benchmark and evaluation)
libraries & packaging application frameworks

Our approach

FP3C

We will study and define programming models and languages to provide an interface between system technologies (ultra-large scale platform, accelerators, fault resilience, power, ...) and application technologies (parallel algorithms, libraries, benchmark, ...).

- Our approach is:
  - To first develop researches on each of the described topics during the first year
  - To study and specify the interface between "system" and "app" during the first 6 months of the project.
  - To integrate several of the proposed solution and begin to program using the proposed language and programming frameworks in 2nd year.
  - The last year, integration of the different software, systems and tools will be proposed.
- Our project presents a vertical stack from applications to low level architecture considerations, in addition to horizontal runtime system researches.
  - Due to the no-dissociable dependences between these issues, it is very important to handle all issues in the same research project.


(parallel algorithms, libraries, benchmark, ...).
application technologies
programming model and Language
system technologies
(ultra-large scale platform, accelerators, fault resilience, power, ...)

FP3C

Project organizations

J1 : Tsukuba (PI)
6 permanent researchers

J2: Tokyo Tech
4 permanent researchers

J3: U. of Tokyo
8 permanent researchers

J4: Kyoto University
2 permanent researchers

F1: INRIA Saclay (PI)
2 permanent researchers

F2: CNRS/IRIT
8 permanent researchers

F3: CEA
4 permanent researchers

F4:INRIA/Bordeaux
6 permanent researchers

F5: CNRS/PRISM
2 permanent researchers

F6: INRIA/Rennes
2 permanent researchers

JFLI
Japan-France Laboratory for Informatics

20 permanent researchers from 4 sites

24 permanent researchers from 6 sites

Research topics based on XMP in FP3C

FP3C

- Extension to GPGPU/manycore
  - XMP concept is to control explicitly where computation is performed
  - Use gmove and loop mapping to core in GPGPU
- Support for fault-resilience
  - collaboration with Franck's team
  - Provide Language constructs to restrict check pointing
  - Communication generation to reduce check pointing

```
#pragma xmp nodes p(*)       // decl of nodes
#pragma xmp nodes gpu g(*)  //decl of GPU cores
…
#pragma xmp distribute AP() onto p(*)  // data
      mapping
#pragma xmp distribute AG() onto g(*)
#pragma xmp align G[i] with AG[i]      //align
#pragma amp align P[i] with AP[i]
int main(void) {
…
#pragma xmp gmove  // transfer CPU⇒GPU)
    AG[:] = AP[:];
#pragma xmp loop on AG(i)
    for(i=0; …)                    // computing on GPU
        AG[i] = ...
#pragma xmp gmove      //  data trasfer(GPU⇒CPU
    AP[:] = AG[:];
```