EUROPEAN COMMISSION

Community research

is-enes
INFRASTRUCTURE FOR THE EUROPEAN NETWORK FOR EARTH SYSTEM MODELLING

IS-ENES2 DELIVERABLE (D -N°: 3.1)

Report on the technology tracking

File name: IS-ENES2_D*3.1*.doc

Author(s): *G. Aloisio, S. Mocavero, P.M Slavin, G.D. Riley*

Reviewer(s): *M.A. Foujols, R. Budich*

Reporting period: *01/04/2013 – 30/09/2014*

Date for review: *18/09/2014*

Final date of issue: *05/06/2015*

Revision table			
Version	Date	Name	Comments
0.1	18/09/2014	Giovanni Aloisio	First formal release
0.2	23/09/2014	Reinhard Budich	Explain the term "interoperability between coupling technologies"
0.3	25/09/2014	Marie-Alice Foujols	Add DYNAMICO informations in chapter 2. Add references for figures when required.
0.4	30/09/2014	Giovanni Aloisio	Final version - reviewers' comments integrated
0.5	03/06/2015	Giovanni Aloisio	Final version with "Conclusion and Recommendations" section
final	5/6/2015	RBudich MA Foujols	Final touches

Abstract

The deliverable aims at summarizing the European and global activities addressing developments both in next generation models including exascale issues and in coupling technologies in order to bring that knowledge into the European climate community. Starting from the analysis of the main exascale issues in climate change research, it provides an overview of the new parallel approaches for the climate models. Moreover, it reports some details about the first attempts of co-design work to establish the interrelationship among physics, algorithms, system software and hardware. Some work towards the interoperability of coupling technologies is also reported. This report is based on the results presented at the "ENES Workshop on Exascale Technologies and Innovation in HPC for Climate Models", held in Hamburg on March, 17-19, 2014 (https://verc.enes.org/ISENES2/events/ws3).

Project co-**funded by the European Commission's Seventh Framework Programme (FP7; 2007**-2013) under the grant agreement n°312979		
Dissemination Level		
PU	Public	
PP	Restricted to other programme participants including the Commission Services	
RE	Restricted to a group specified by the partners of the IS-ENES2 project	
CO	Confidential, only for partners of the IS-ENES2 project	

CAPACITIES

Table of contents

Executive Summary

Earth System Models (ESMs) simulate a large variety of processes on a variety of time (from seconds to centuries) and space scales (from 1 to 100s of km) for global or regional areas. They are a canonical example of multi-physics and multi-scale modeling. The system is physically characterized by sensitive dependence on initial and boundary conditions and natural stochastic variability. Very long integrations are needed to extract signals of climate change. Since computational cost increases nonlinearly with higher resolution, it is known that high fidelity climate simulations at 1 km resolution will require extreme scale computers. However, I/O and memory-bound, multi-physics codes present particular challenges to computational performance. Indeed, most of the legacy and even some of the current advanced climate applications require code re-engineering, new parallel model design and new algorithms, to exploit new and emerging computational architectures and to increase spatial and time resolution at reasonable cost. It is necessary to design scalable computational kernels and algorithms, as well as considering new approaches and paradigms which are better suited to exploit high levels of concurrency found in parallel HPC systems. A co-design approach is suggested, allowing scientific experts from the application domains, including computational scientists and software engineers, as well as mathematicians to work together on the scientific problem tightly with the technology developers. This will speed-up the development of models and their use on future exascale computers, improve the efficiency of the modeling community and the dissemination of model results to a large community of users.

The report presents the research efforts in numerical climate sciences towards exascale and some results encompassing a variety of current leading-edge computing architectures. The principal programming model is the hybrid MPI+X. However, novel methods, including OpenACC, are also described. The general lesson is that extracting performance from novel architectures remains a challenge, and institutions would benefit from substantially larger investments in software in relation to hardware than in the past, following a co-design strategy. This unfortunately does NOT mean that investments into hardware can be reduced!

Developments in coupling technologies are also required for next generation models. There is a growing interest in the issue of interoperability between coupling technologies as coupled models are increasingly developed as multi-institution collaborations, with each institution bringing specialized models which may be built with different coupling technologies[1]. Some work on using different technologies within the context of a single model is presented.

[1] In the U.S. there is current interest at NOAA in coupling models developed using separate technologies such as the Basic Modelling Interface (BMI) in the Community Surface Dynamics Modelling System (CSDMS to models developed using the NUOPC interface of ESMF. In Europe, OASIS is now integrated with the U.S Model Coupling Toolkit (MCT). At UNIMAN, we are exploring extending our Bespoke Framework Generator (BFG) to support the exchange of coupling data between models which use different frameworks, such as ESMF and OASIS, through the generation of suitable 'framework adaptor models' that mediate the exchange of data between frameworks by being able to 'speak' the language of both; the work reported here is in this context.

This report is based on the results presented at the "ENES Workshop on Exascale Technologies and Innovation in HPC for Climate Models", held in Hamburg on March, 17-19, 2014 (https://verc.enes.org/ISENES2/events/ws3).

1. Towards Exascale: research in climate science

Numerical climate change research needs the integration/coupling of several models both for high-resolution simulations for short-term prediction and for long-term climate projection. In the future, exascale are expected to be able to provide the computational resources needed to increase resolution, complexity and ensemble size with sufficient efficiency. Simulation duration will benefit from exascale only if models will be able to scale. The science drivers towards exascale in climate modeling can be classified as the need for larger ensembles of simulations, higher spatial resolution, increased complexity of processes represented, and longer timescales simulated. None of the drivers promises strong scalability. At most, we expect weak scalability, most clearly for ensemble size and spatial resolution. Scalability for increased complexity is unclear, while for longer simulations it is non-existent with current time-stepping algorithms. Computational scalability is easiest to reach for ensembles; however, output organization of large ensembles is highly nontrivial. Moreover, the slower reduction in the cost of storage relative to computing creates an ever more pressing need to re-think current data storage strategies. Hence, there are several challenges for the current climate models to face when scaling at an order of 10^{18} threads, as required at exascale. This section presents many European initiatives which aim to face these challenges. It also includes a view on two of the main challenges at exascale: the data volume management and the standardization of performance evaluation.

1.1 Climate modeling at exascale

a) G8 ECS (Enabling Climate Simulations at Extreme Scale) project

G8 ECS is a three-year international collaboration between Spain, Germany, France, the United States, Canada and Japan. The project focuses on three issues thought to be critical for efficient use of exascale systems by climate models: node performance, scalability and resilience. Two models are the subjects of the study: the U.S. Community Earth System Model (CESM) and the Japanese Non-hydrostatic Icosahedral Atmospheric Model (NICAM). Specifically, the main scientific results from the project are:

1) New methodology, tools and techniques to discover and mitigate scalability and performance bottlenecks. Tools like *Extrae* and *Paraver*, both from the Barcelona Supercomputing Center (http://www.bsc.es/computer-sciences/performance-tools/paraver), as well as *Scalasca*, from Jülich Supercomputing Center (http://www.scalasca.org) revealed many unknown performance issues in CESM.
2) Advanced communication avoiding algorithms and tasks load balancing environment for efficient parallel executions on hybrid node architectures.
3) Innovative low overhead resilience techniques for local tolerance of process failures and silent data corruptions.

More information about algorithms and techniques can be found in the "G8 Enabling Climate Simulation at Extreme Scale" final report available at the following link: https://wiki.cites.illinois.edu/wiki/download/attachments/479134162/G8-ECS-final-report.pdf?version=1&modificationDate=1403286885000&api=v2.

b) CESM refactoring

The National Center for Atmospheric Research (NCAR) is working on the refactoring of CESM for exascale starting from the consideration of some of the climate use cases that are directly relevant to exascale. Following the climate models evolution trend (Figure 1), each case involves different resolutions and experimental ensemble configurations that range from very few instances of experiments spanning a large portion of the system (ultra-high resolution case) and having a high vulnerability to system fault, relatively low data volumes and large individual file sizes, to the very large ensemble experiments with high resilience to faults, relatively large data volumes and small individual file sizes. Climate predictability experiments (data assimilation) are slotted in between these two extremes.

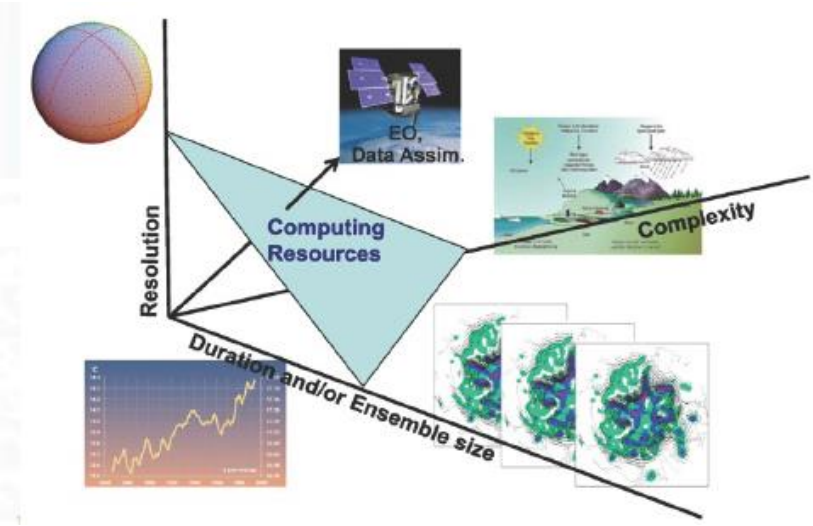


Figure 1 - Climate models evolution trend towards exascale (image of Jim Kinter)

The development of the new Spectral element-based atmospheric dynamics (dycore) permitted scalable CESM performance at high resolution (Figure 2). Further improvements

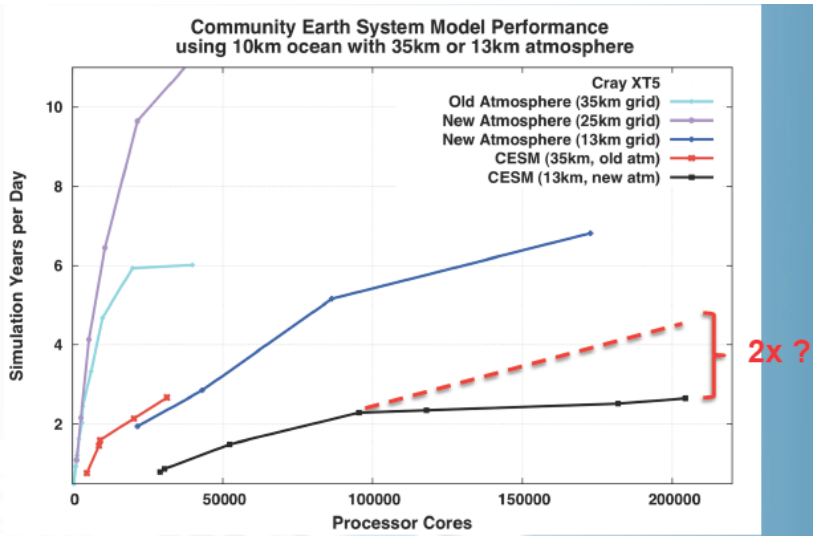


Figure 2 - Performance comparison with/out the new atmospheric dycore (image of Pat Worley)

require optimization of ocean and sea ice models.

Moreover, the use of accelerators (MIC and GPUs) and stacked memory to run the ultra-high resolution case, tested on the Disconinuous Galerkin Gradient Kernel, allows an improvement of the performance results (Figure 3).

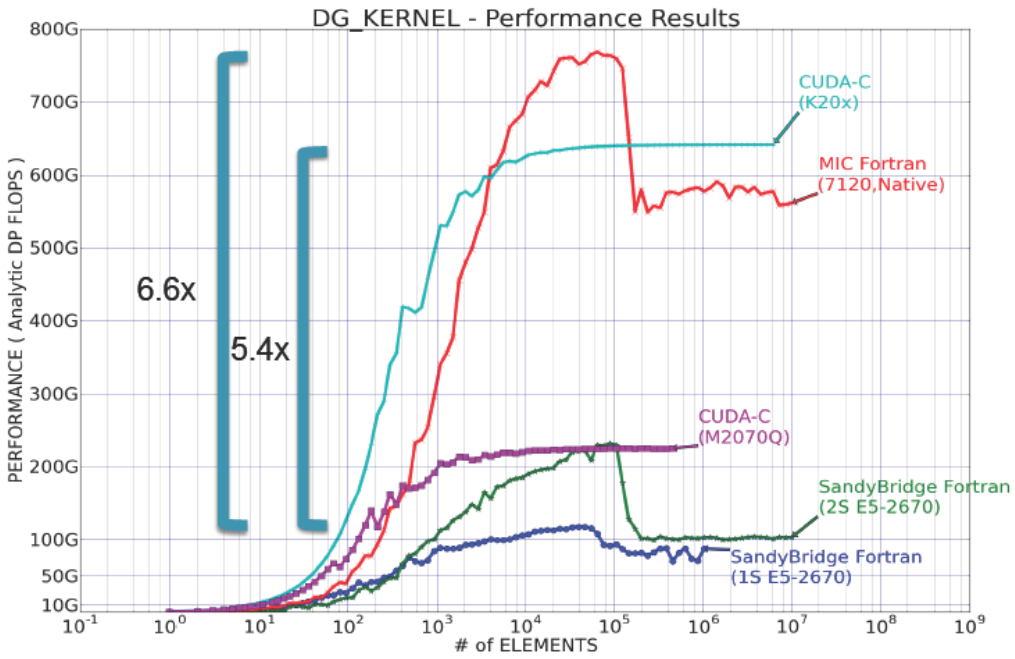


Figure 3 - Performance results comparison with/out using accelerators

Robust parallel algorithms for data assimilation, probably based on some kind of ensemble forecasting, to do climate prediction at exascale, are needed.

The NCAR exascale strategy aims at:

- Computational optimization. A complete redefinition of the current code enhancement methodology, too slow for exascale, is needed. The new methodology, reported in Figure 4, is based on the automatic generation of unit test, the use of robust profiling and debugging tools, the SSMA (Single Source Multiple Architecture) programming paradigm.
- Data optimization. The current system storage cost balance between compute and storage (currently about 80% computing and 20% disk/analysis systems) will be as much as 10x off by the end of the decade, meaning that unless we change how we do business, storage costs will swamp compute costs long before we reach the capability to deploy an exascale platform dedicated to climate simulation. Lossy compression of climate data could save a factor of five in storage costs, partially restoring this balance.

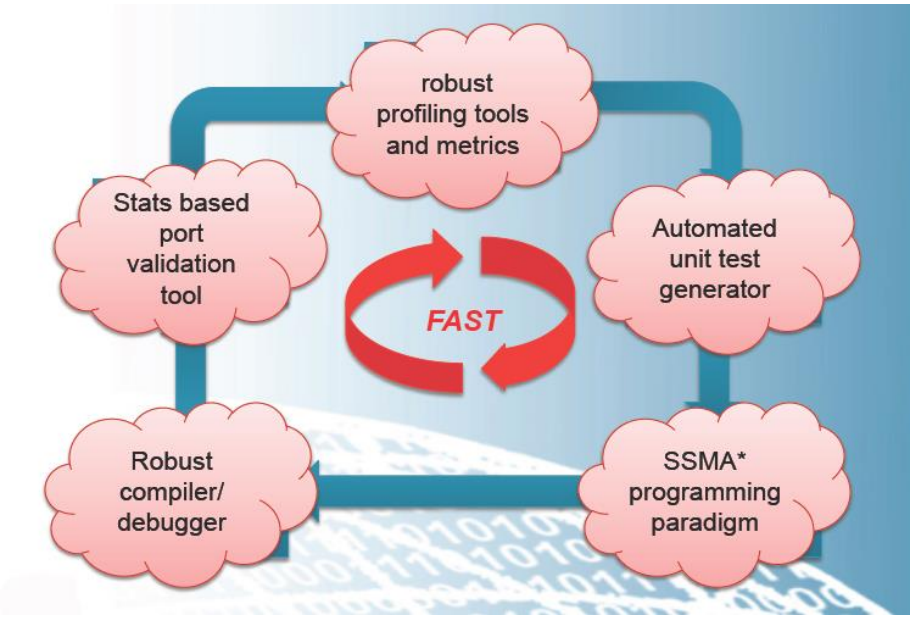


Figure 4 - Performance Enhancement Methodology: a virtuous cycle for code improvement (*single source multiple architecture)

c) ICOMEX (ICOsahedral-grid Models for EXascale Earth system simulations) project

The ICOMEX G8 project is a collaboration between Germany, France, UK and Japan. It addresses roadblocks towards exascale computing in four advanced earth system models: NICAM, ICON, MPAS and DYNAMICO.
While these models are all based icosahedral grids, they differ in terms of numerics and grid structure. Within the project, firstly, the scientific accuracy and hardware performance of the models are compared, secondly, key issues which are performance hurdles in computation and I/O are pushed.
In detail, the following work is covered:

1. Intercomparison of model performance.
   This covered meteorological aspects, e.g. the Baroclinic wave test, as well as computational performance and scaling on the K computer.
2. An evaluation of implicit solvers for icosahedral grid models.
   MPAS has been extended by a Strang-Carryover scheme and multigrid Helmholtz solver to evaluate the benefit of implicit solvers. The performance is under evaluation.
3. A bottom-up approach to investigate kernels with alternative memory layout to derive best practices for the model development to increase programmability and performance portability.
   A light-weight extension to Fortran allowed the modeller to operate with high-level objects such as Cell3D. A source-to-source compiler implemented in the ROSE compiler infrastructure has been implemented which translates the DSL into Fortran code.

During this process, the memory layout and loop order of multi-dimensional data structures has been created according to an architecture-specific configuration. For ICON, a well tuned memory structure increased performance by 20%.

4. Parallelization of internal post-processing.

   The main point is not to parallelize existing post-processing methods but to develop new algorithms with better scaling behavior in terms of computational complexity and parallelization.

   A 2nd order scalable conservative remapping scheme has been developed which outperforms SCRIP significantly. The resulting library has been integrated into XIOS.

5. Analysis and optimization of the I/O stack.

   A bottom-up analysis of I/O performance revealed several shortcomings in the I/O stack. On DKRZ supercomputer, a developed patch for NetCDF removed the additional caching inside the library yielding 3x improvement.

   A prototypical implementation of an alternative HDF5 file representation achieved 10x speedup in parallel writes (close to peak performance) and shows the benefit of a system-specific file layout.

   Additionally, application-specific compression have been investigated. With MAFISC a new lossless compression pre-conditioner has been developed which achieves better compression rates than uninformed state-of-the-art compression schemes. Lossy compression has been briefly investigated comparing accuracy of GRIB2 and APAX.

6. Collaboration with hardware vendors.

d) LFRic programme

LFRic is a new programme of work to deliver, towards the end of this decade, a replacement for the Met Office Unified Model that is more scalable on future HPC systems. LFRic is based around the "Gung Ho" dynamical core being developed in collaboration with the UK academic community and STFC Daresbury. Gung Ho offers a number of challenges to developing a modeling infrastructure. It has a Finite Element formulation on a semi-unstructured grid, which does not fit many of the software tools used for existing climate and weather modelling systems, and it presents new scientific challenges for coupling to Finite Difference model components.
LFRic is characterized by three phases of development:

- 2016: Gung Ho dynamics with a computational infrastructure
- 2019: First version of the atmosphere model

- 2022: Operational deployment

LFRic and Gung Ho are researching ways of separating the concerns of a scientist developing code to implement algorithms and kernels from the concerns of a computational expert implementing the code required to run the model efficiently on a highly parallel system (Figure 5).

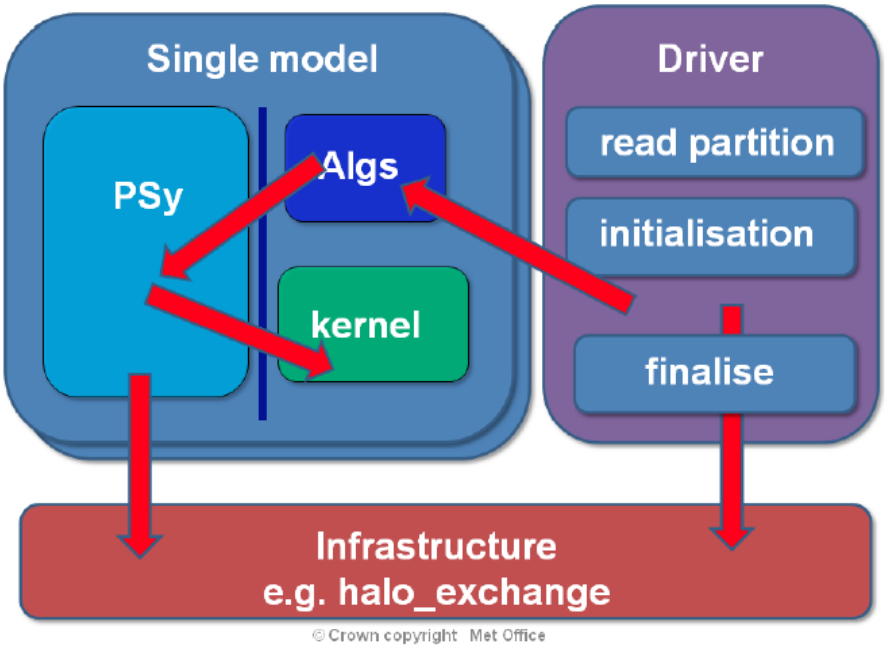


Figure 5 - Separation of concern between Parallel Systems (the PSy layer), and Algorithms and Kernels.

e) SPECS project

The FP7 SPECS project aims to produce quasi-operational and actionable seasonal-to-decadal, local climate information with a new generation of reliable European climate forecast systems, including initialised ESMs, efficient regionalisation tools and combination methods, and an enhanced dissemination and communication protocol over land, with a focus on Europe, Africa and South America. Several partners in the SPECS project make use of PRACE resources to perform some of the most challenging climate-prediction experiments undertaken to date. One of these projects is HiResClim, which, among other climate simulations, is producing seasonal and decadal climate predictions over the period 1993-2009 with two different high-resolution models: EC-EARTH and CNRM-CM. The interest of this kind of experiments for large HPC platforms is that they require both capacity and capability because the jobs scale up to several thousand cores while many independent jobs can be run at the same time, making an optimal use of the machines and reducing the usually very long time to response encountered in climate modeling problems. However, managing these large problems require adapted workflow and monitoring tools that can efficiently and flexibly make use of the largest machines available. Autosubmit, a tool developed at IC3 in the

framework of the FP7 IS-ENES2 project, responds to this need and makes the task of setting up and performing these experiments particularly transparent to the user. Preliminary results show that the increased model resolution of the SPECS experiments benefits the quality of both the mean climate simulated and the quality of the forecasts in ways that are not trivial.

f) DYNAMICO project

The DYNAMICO project aims at the development of a new dynamical core for LMD-Z, the atmospheric general circulation model (GCM) part of IPSL-CM Earth System Model.
LMDZ4, the current version of LMD-Z, has a shallow-atmosphere, hydrostatic dynamical core. It is based on a latitude-longitude C-grid, a hybrid pressure-based terrain-following vertical coordinate, second-order enstrophy-conserving finite-difference discretization and positive-definite advection. Grid refinement is implemented as a continuous zoom via smooth grid stretching. An extensive package of physical paramererizations is coupled to the dynamical core.
The primary goal of DYNAMICO is to re-formulate in LMD-Z the horizontal advection and dynamics on an icosahedral grid, while preserving or improving their qualities with respect to accuracy, conservation laws and wave dispersion. In turn, a new grid refinement strategy is required. A broader goal is to revisit all fundamental features of the dynamical core, especially the shallow-atmosphere approximation, the vertical coordinate and the coupling with physics. Efficient implementation on supercomputing architectures is a key issue addressed by DYNAMICO. The hybrid implementation (MPI/OpenMP) and the use of XIOS to manage the output writing allow to achieve the performance improvement shown in Figure 6.

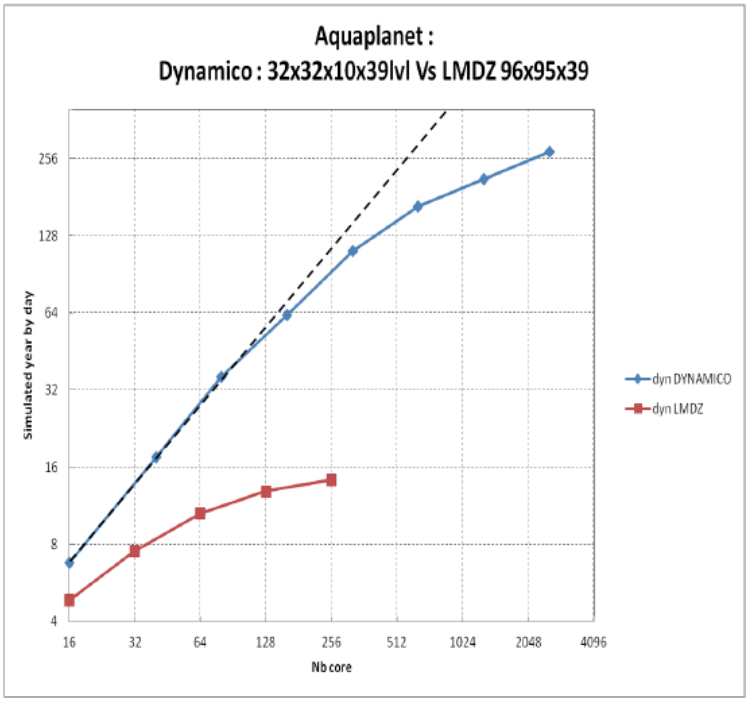


Figure 6 - Performance comparison between DYNAMICO and LMD-Z

1.2 Data issues at exascale

One of the main issues in climate sciences is the quantification of uncertainty and it's reduction, requiring to increase model resolution. Increasing the resolution also opens severe issues in extreme data management like mass data storage, and dissemination of model outputs for analysis to a wide-ranging community of scientists over a long period, which will need to be addressed.
There is an increasing need to store significant research data in a distributed environment for efficient access and disaster recovery. The problem has been that storing complete replicas in multiple sites is costly from a standpoint of both storage hardware and power. There are several commercial solutions, such as the WOS (Web Object Scaler), developed by the DDN Company. It is an automated, high performance data distribution and protection mechanism aiming at satisfying the storage requirements in terms of reliability, performance, scalability, accessibility and efficiency. WOS is used to store data for space science, bio-informatics, climate change and the paleo-sciences. A model has been developed that allows geographic data distribution without the need of complete replicas, offering high reliability and quality of service. A mathematical model is used to distribute research collections safely and efficiently to three locations for example using the aggregate storage space of less than two times the space needed for the original data. However, the need to manage extreme data requires the definition of a shared strategy to be adopted at European level.

a) ExArch project

The ExArch (Climate analytics on distributed exascale data archives) project is a collaboration among British Atmospheric Data Centre (UK), Princeton University of Toronto (CDN), IPSL (FR), DKRZ (DE), UCLA (US) and CMCC (IT). It aims at developing a strategy, prototype infrastructure and demonstration usage examples for scientific analysis of exascale archives. ExArch will provide software to enable the global federation to meet exa-scale demands. The main goals are:

1) Efficient data discovery and robust provenance handling. Advanced search and data access services of the distributed archive are exploited to interrogate remote and local archives and receive a scientifically relevant answer that is structured in components to reduce the volume of data required to be transferred. A structured approach to storing query responses, mirroring the data structure within the archive, ensures re-usability of responses.
2) Machine readable technical documentation of climate models. Developed technical schema to document the climate models used in the IPCC 5th Assessment Report. ExArch has provided flexible programming and browsable interfaces to allow users to explore and compare these.
3) Complexity in data processing. ExArch has brought together teams working on a variety of different approaches to the analysis of large climate data archives. The huge and rapidly expanding complexity of the data feeds into a complex range of processing

options. ExArch aims facilitating the exploitation of synergies and complementarities of these approaches.

1.3 Performance evaluation

Traditional metrics of computational efficiency such as performance counters and scaling curves do not tell us enough about real sustained performance from climate models. They also do not provide a basis for comparative information across models.
The novel idea is to define a set of metrics that can be used for the study of computational performance of Earth System models. These measures do not require specialized software or specific hardware counters, and should be accessible to anyone. They are independent of platform, and underlying parallel programming models.

These metrics take into account the following considerations:

1) Models can have two optimal points: one for speed (minimizing time to solution, maximizing simulated years per day or SYPD), the second for best use of a resource allocation (minimizing compute-hours per simulated year, or CHSY).
2) ESMs generally are configured to run more than one component concurrently: it is needed to measure load balance and coupler cost.
3) Computational cost scales with the number of degrees of freedom in the model. This number can be factorized separately into resolution (number of spatial degrees of freedom) and complexity (number of prognostic variables). This separation is useful because performance varies inversely across resolution and complexity in weak-scaling models.
4) Codes are memory-bound: memory bloat can be measured by comparing actual memory usage, to the theoretical minimum implied by the resolution and complexity above.
5) Models configured for scientific analysis bear a significant I/O load (can interfere with optimization of computational kernels). Data intensity (GB/CH) is a useful measure for designing system architecture.
6) Actual SYPD (ASYPD) measure the SYPD achieved from a long-running model, as opposed to a single segment. A significant difference here indicates the need to devote resources to system and workflow issues rather than optimizing code.

2. Overview of the new parallel approaches for climate models

2.1 Hybrid programming

All large-scale parallel computers now and for the predictable future will be assembled of shared-memory nodes on a high performance interconnect. While many applications still use the message-passing interface for all parallelism, hybrid programming models, which combine several parallel programming models or systems in the same program or in different components or routines, each of which is in a single parallel programming model, are becoming increasingly popular. The aim is to address issues such as (i) declining memory per core; (ii) having multiple threads/core; (iii) improving load balance and (iv) algorithmic issues. Pros of hybrid programming is related to the management of (i) compute-bound loops (many operations per load from memory), (ii) memory bound loops (where read data is shared, so that cache memory can be used more efficiently), (iii) fine-grain parallelism (algorithms that require frequent exchanges of small amounts of data) and (iv) the load balancing due to the ease of moving data/tasks.
However, sometimes pure MPI model is better: trying to use an hybrid approach on very regular, memory-bandwidth-bound computations is likely to loose because of the better, programmer-enforced memory locality management in the pure MPI version.
The placement of processes and threads is critical for performance: placement of processes impacts use of communication links (poor placement creates more communications), while placement of threads within a process on cores impacts both on memory and intranode performance. Parallel programming models need to provide ways to coordinate resource allocation, taking into account the numbers of cores/threads/functional units, the affinity of cores/threads, intranode and internode memory bandwidth. They must also provide clean ways to share data. Developers must take into account performance issues, dialoging with standard community (i.e. OpenMP, MPI committees). Growing complexity of code will require adopting approaches that distance developers from the final code. The central principle is the division of the code into layers to separate the natural science and computer science aspects. The identification of these layers enables a "separation of concerns" between largely independent components which have different requirements and need developers with very different skill sets.

In the HD(CP)^2 project, some experience was gathered using the hybrid programming approach. In order to reduce significantly the uncertainty of climate change projections, it is crucial to understand cloud and precipitation processes. Within the HD(CP)^2 project an LES model based on ICON (ICOsahedral Non hydrostatic GCM) is being developed. The ICON-LES model aims at resolving cloud and precipitation processes using grids with a resolution of 10000x10000x400 grid elements and a grid spacing of 100m. Such simulations are computationally very intensive. Although, modern high performance computing platforms can deliver a peak performance in the range of Petaflop/s, ICON is, despite it's quit a modern layout, not yet able to exploit this power. The two major bottlenecks are communication (i.e.

moving data between the levels of the memory hierarchy inside nodes/cores or among processors), and fast parallel I/O.
Avoiding communication by increasing temporal and spatial locality is a key issue. The use of an Hybrid MPI/OpenMP parallelization is one of the feasible approaches to address it. In this way MPI communication has been reduced and ICON achieves an efficiency of 85% on 32000 cores and 73% on 65000 cores, as shown in Figure 7, when using very limited output.

Figure 7 –Strong scaling of ICON model at higher resolution using an hybrid parallelization approach

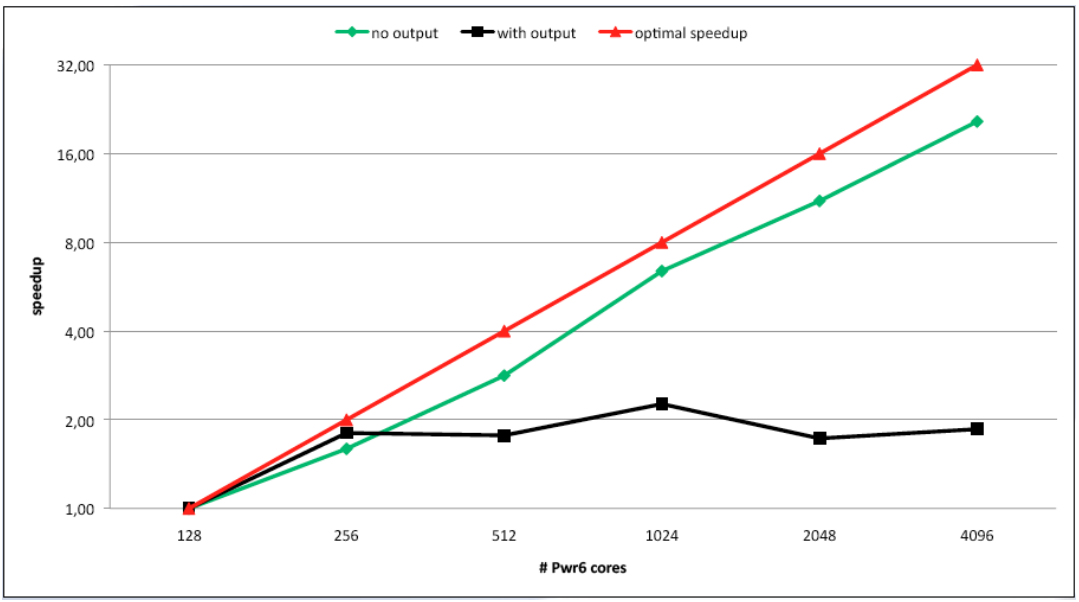


Figure 8 – Impact of Serial I/O for LAM416m on BLIZZARD

The efficiency drops significantly when activating full output, which shows that the I/O problem remains a real challenge. Figure 8 shows the impact of Serial I/O for LAM416m on BLIZZARD, the current DKRZ production machine (IBM P6, ~8000 cores, see https://www.dkrz.de/Klimarechner-en/hpc/ibm-en)

2.2 Use of coprocessors in climate models

Numerical Weather Prediction (NWP) and Climate models represent an application domain, which is capable of exploiting, and requires the use of HPC Exascale resources. Intel's Xeon Phi coprocessor is an example of architecture, which could be a prototype for Exascale HPC machines. In particular, it has relevant features such as a high degree of parallelism and heterogeneity of host and device processors and distinct memory spaces.
The Met Office's Unified Model (MetUM) is a mature, fully featured, NWP and Climate model. The experience of porting the UM to the Xeon Phi for the third generation of dynamical core, called ENDGame, has been carried out. The dynamics code, especially the solver required to solve the Helmholtz equation takes a significant portion of runtime, thus the performance of this component is an important consideration. Two different data layouts - lexicographical data layout (lcg) and red-black data layout (rb) - and three data volumes - small (typical for global simulation), medium (typical for regional simulation) and large for unfeasible large simulation, have been taken into consideration. The architectural features of the Xeon Phi have been employed. Experiments with code transformations aimed at boosting the performance on the Xeon Phi (KNC) are evaluated and compared to other processors such as Intel and IBM CPUs (Sandy Bridge (SB), Ivy Bridge (IB) and Power7 (PWR7)). Scalability increases with the data volume, as shown in Figure 9.

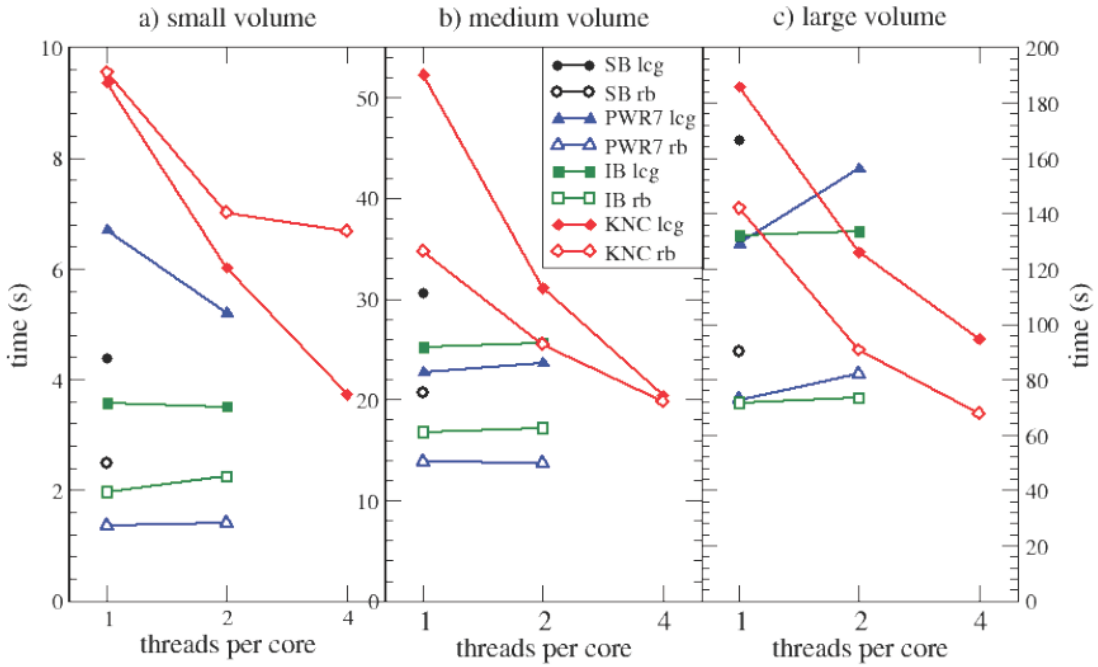


Figure 9 – Performance comparison among Xeon Phi and Intel processors

Moreover, the cubed-sphere dynamical core, described in Figure 10 and used by NASA/Goddard and NOAA/GFDL, has been ported on the Intel Xeon-Phi multi-core architecture, equipped with coprocessors based on "Many Integrated Core" (MIC). Performances have been compared with Xeon-SNB. The dynamical core is a finite-volume method implemented on a cubed-sphere grid and represents a significant portion of the computational time used in simulations run at both organizations.

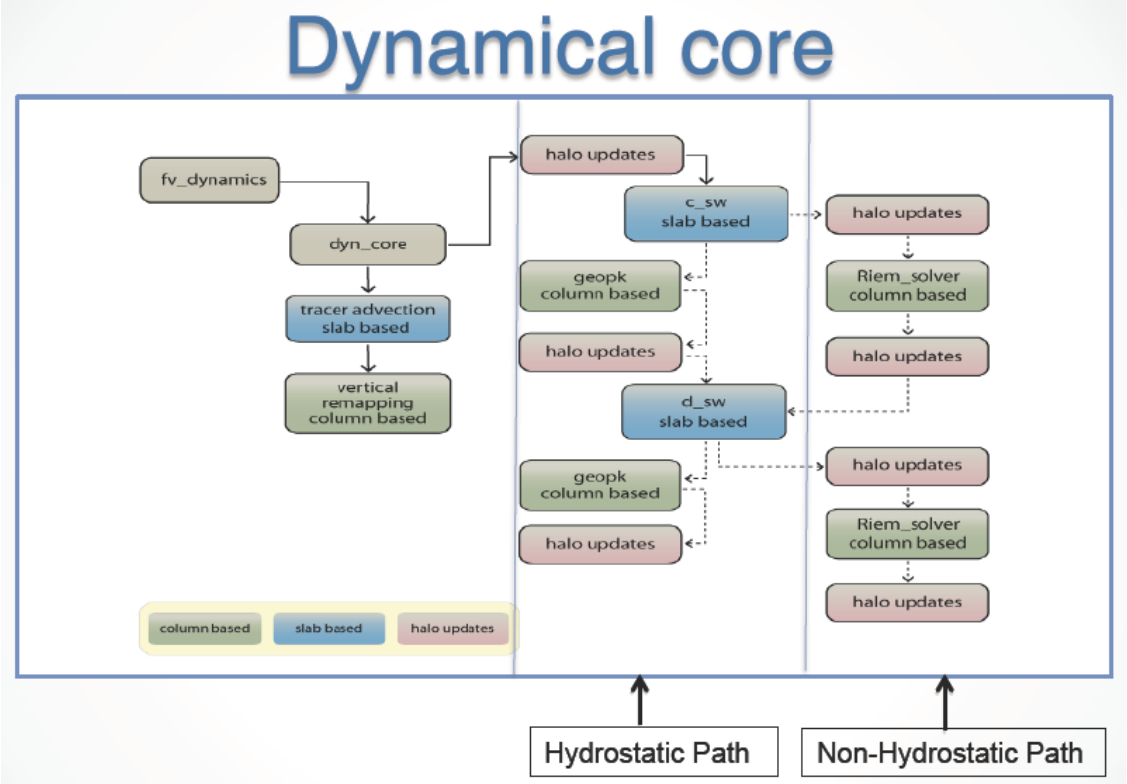


Figure 10 - Cubed-sphere dynamical core used by NASA/Goddard and NOAA/GFDL

The hybrid parallelization strategy implies the block decomposition of the generic MPI subdomain, as shown in Figure 11.

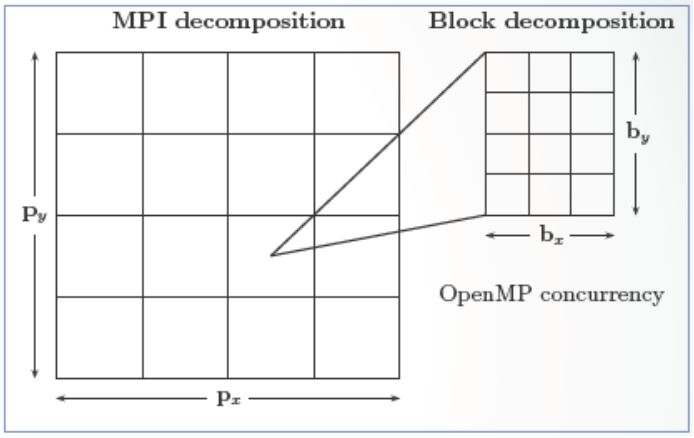


Figure 11 - Hybrid parallelization domain decomposition

Two different concurrency approaches are considered respectively for slab and columns routines, as shown in Figure 12. Performance on the Xeon-Phi requires: efficient MPI communication to handle halo updates, large-scale OpenMP concurrency to satisfy the available 240-threads (60-cores with 4 HW/core), strong single-core performance maximizing vectorization, data-alignment, and data reuse. Additionally, to result in a useful code-base for scientific experiments, it is needed to maintain a single source for both conventional Intel: Xeon and multi-core Xeon-Phi architectures. Additionally, the code-base should have no performance degradation on conventional architectures and the software should be readable for the scientists.
Results show performance degradation on Xeon-Phi compared with Xeon-SNB. The code needs invasive refactoring to improve performance. Moreover, the code has a flatter performance profile, so the use of kernels to predict performance can be misleading.

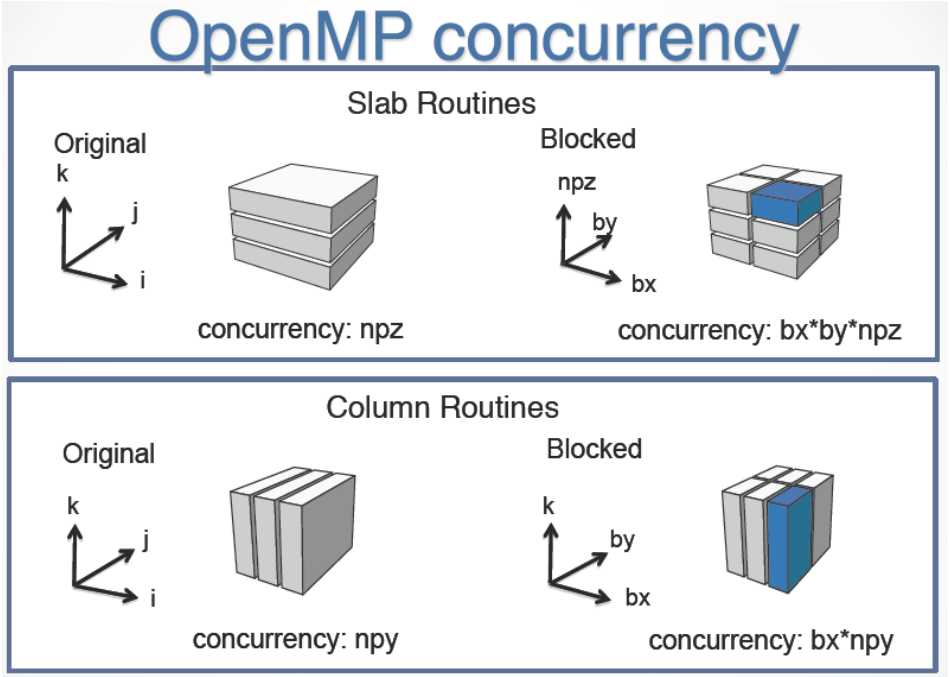


Figure 12 – Slab and column concurrency approaches

2.3 Use of accelerators in climate models

In the future, increased performance per energy cost will be the key driver behind scientific decisions to develop Earth System Models for hybrid GPU-based architectures: NVIDIA is working towards Exascale HPC direction through the implementation of GPUs and related technologies (Figure 13).

Figure 13 – NVIDIA exascale directions

Tesla GPUs advantages are:

- Power efficiency.
- Ease of programming and portability.
- Growth in application availability.

The programming strategy follows three different approaches with increasing development effort:

- Use of GPU libraries which provides a fast "drop-in" acceleration,
- The adoption of OpenACC directives which guarantees an acceleration maintaining the standard language (Fortran, C, C++),
- Use of a different programming language which allows to maximize the GPU architecture flexibility.

The growth of GPU accelerated applications, shown in Figure 14, is led by the climate applications shown in Figure 15.

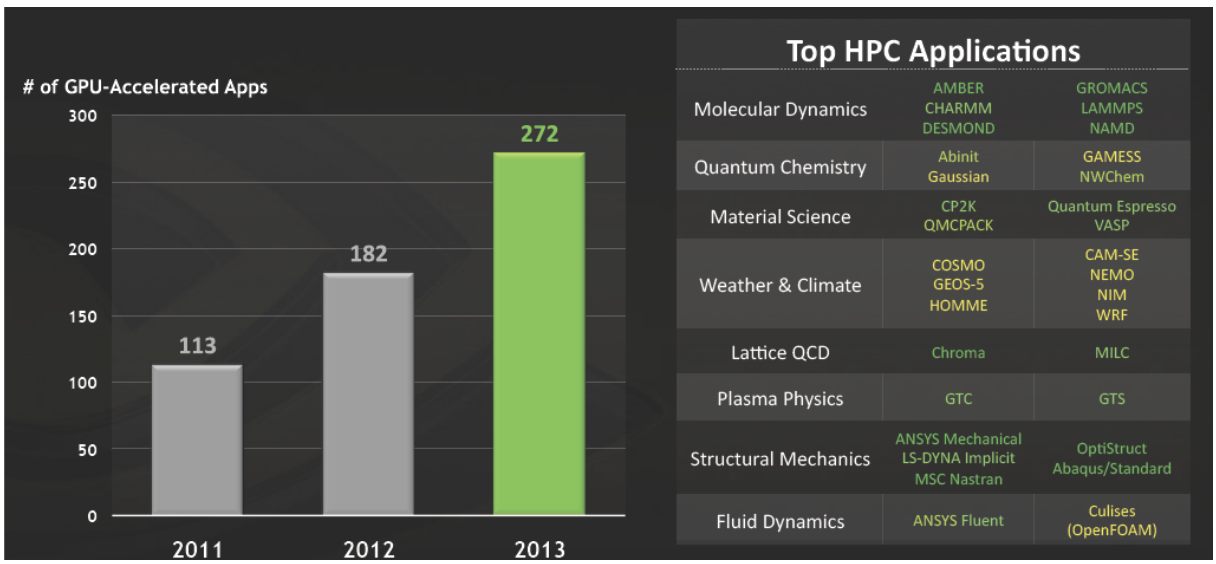
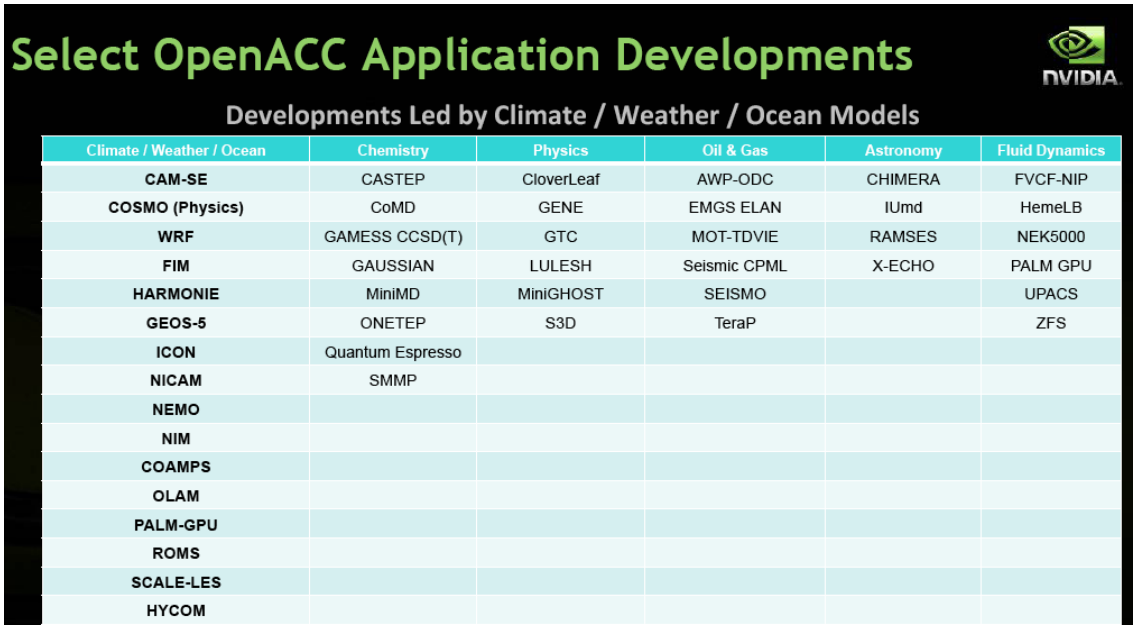


Figure 14 – Growth of GPU accelerated applications



Climate / Weather / Ocean	Chemistry	Physics	Oil & Gas	Astronomy	Fluid Dynamics
CAM-SE	CASTEP	CloverLeaf	AWP-ODC	CHIMERA	FVCF-NIP
COSMO (Physics)	CoMD	GENE	EMGS ELAN	IUmd	HemeLB
WRF	GAMESS CCSD(T)	GTC	MOT-TDVIE	RAMSES	NEK5000
FIM	GAUSSIAN	LULESH	Seismic CPML	X-ECHO	PALM GPU
HARMONIE	MiniMD	MiniGHOST	SEISMO		UPACS
GEOS-5	ONETEP	S3D	TeraP		ZFS
ICON	Quantum Espresso				
NICAM	SMMP				
NEMO					
NIM					
COAMPS					
OLAM					
PALM-GPU					
ROMS					
SCALE-LES					
HYCOM					

Figure 15 – OpenACC application development

NVIDIA is working on GPU hardware and CPU host platforms, but also on system software with emphasis on development tools and compilers for a Fortran programming environment.
The effective use of Graphics Processing Units (GPUs) poses significant programming challenges in Climate Modeling. It can significantly reduce time-to-solution and energy-to-solution, it can allow higher model resolutions and could potentially reduce acquisition and operation costs of new production platforms. The performance profiles of such models indicate that there is no single kernel, which can be offloaded to the accelerator to drastically improve performance. Moreover, there are extensive data dependencies between kernels through three-dimensional fields. Copies of these inside the time-stepping loop would require excessive communication through the PCI-X bus, which would dominate over any improvement the GPU could offer. Thus, porting these models to GPUs implies an "all-or-nothing" strategy, requiring all model components within the time loop to be ported, and data copied to/from the device outside the time loop.
Still, many international teams have stepped up to the challenge. The following eight atmospheric models are being ported to GPUs. For each model, a description statement, some details about the porting status, and the latest results, are reported.

a) CAM-SE (Homme)

Model type: global, cubed-sphere grid, spectral element, hydrostatic
Status: various GPU-enabled kernels available

- Spectral element dynamics (HOMME), in progress
- Tracer advection, implemented
- RRTMGPU, implemented

Production on GPU systems: premature

Figure 16 reports a comparison of performance between Tesla M2050 and AMD Opteron of some GPU-enables kernels.

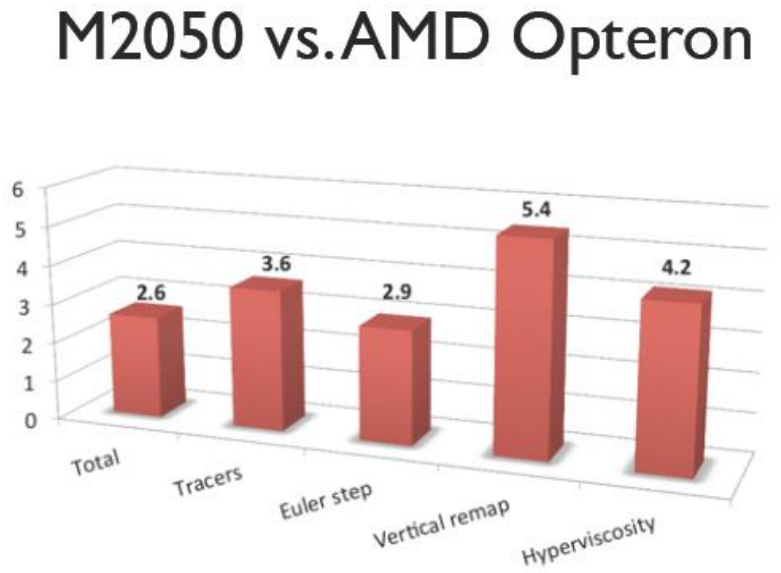


Figure 16 - From: Norman, et al., Titan Workshop ORNL, 2012

b) WRF

Model type: mesoscale, Cartesian grid, non-hydrostatic, Arakawa C-grid, 3-order RK time integration

Status: various GPU-enabled kernels available

- Single-moment 5-tracer microphysics (CUDA-C)
- Double-moment 6-class microphysics
- Fifth order tracer advection
- WRF-Chem chemical kinetics
- Goddard shortwave radiation scheme (CUDA-C)
- RRTMGPU: LW (CUDAFortran) and SW (OpenACC) radiation

Production on GPU systems: no

Some preliminary results are reported in Figure 17 (speedup w.r.t single core) and Figure 18.

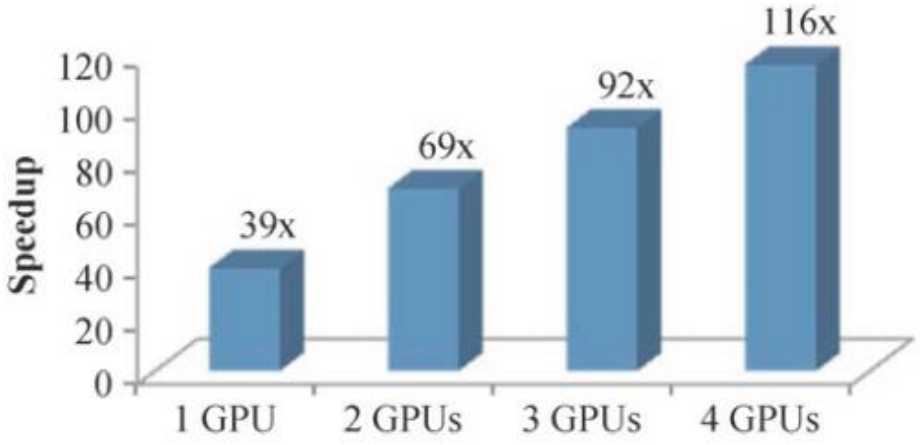


Figure 17 - Goddard SW (Mielikainen, et al., 2012): w.r.t. single core

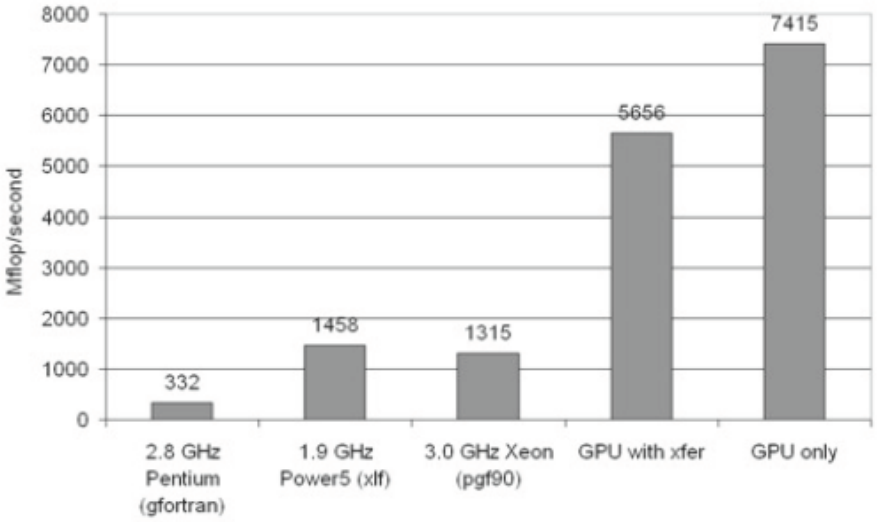


Figure 18 - WSM5 (Michalakes et al., 2009)

c) NICAM

Model type: global, icosahedral, non-hydrostatic
Status: Dynamical core
Paradigm: OpenACC, CUDA
Production on GPU systems: premature
A comparison between the implementation with CUDA and OpenACC are reported in Figure 19.

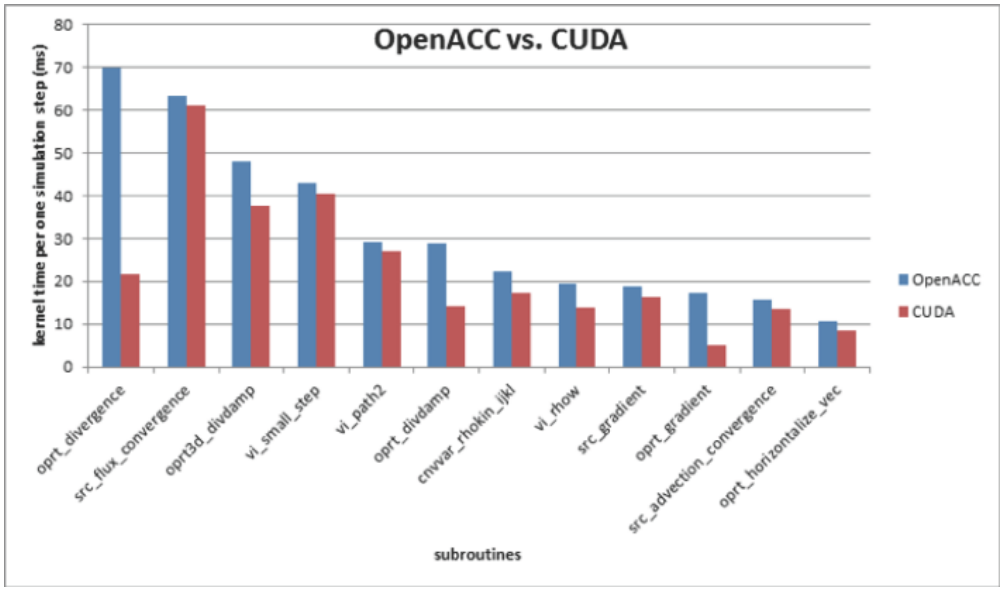


Figure 19 - From: Akira Naruse, Feb 2014

d) ICON

Model type: global, icosahedral (triangular) grid, nonhydrostatic, Arakawa C-grid
Status: dynamical core (testbed version)
Paradigm: OpenACC (multi-node), OpenCL and CUDAFortran (single-node)
Ongoing: consolidating dycore implementation into the trunk
New: port of ICON Physics (with Markus Wetzstein)
Some results are reported in Figure 20.

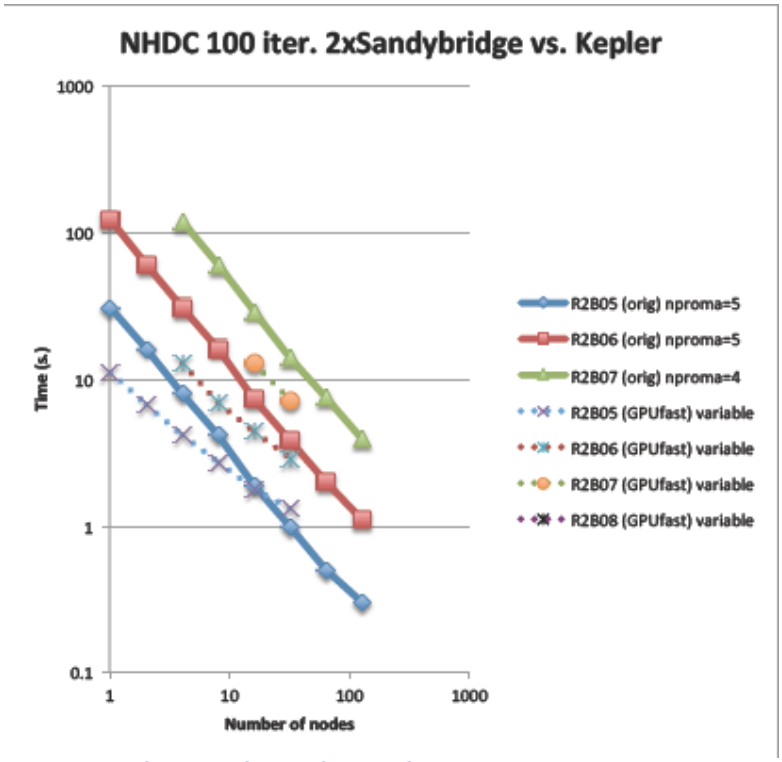


Figure 20 - Compare original (GNU) on Cray XC30 (2x Sandybridge sockets) vs. XK7 node with Kepler K20x (Cray CCE)

e) GEOS-5

Model type: global, finite-volume, cubed-sphere grid, nonhydrostatic dynamics, single and double precision
Paradigms: CUDA/C, CUDAFortran, OpenACC, ESMF
Some results are reported in Figure 21.

Kernel	Speedup (v. Socket)
GWD	5.1x
TURBULENCE	3.0x
CLOUD	2.3x
IRRAD	3.5x / 4.4x
SORAD	4.6x / 6.4x

(8-core SNB vs. K20x)

Figure 21 – Compare 8-core SandyBridge with Tesla K20x

f) FIM/NIM

Model type: global, hexagonal/pentagonal NWP, hydrostatic and non-hydrostatic dycores
Paradigm: MPI + OpenACC (or similar) directives
Status: dynamics + physics. Developed also directives and F2C-ACC compiler.
Some results are reported in Figure 22.

NIM	Opteron	Westmere	SandyBridge	Fermi	K20x
runtime	143.0	86.8	60.0	25.0	20.7

- Parallel performance
  - Being run on up to 160 GPUs
  - Working on optimizing inter-GPU communications

21%

3x

Compiler	Trcadv1	Trcadv2	Trcadv3	Total(% slower)
F2C-ACC	1031	463	679	2173
PGI	1615	885	871	3371 (55%)
Cray	2895	757	1233	4885 (124%)
Cray-fast32	1205	607	1076	2888 (32%)

Figure 22 - From: Mark Govett, Nov 2012

g) GRAPES

Model type: global/regional, non-hydrostatic, semi-implicit, semi-Lagrangian

Status: various GPU-enabled kernels available

- Helmholtz solver, Generalized Conjugate Residual (CUDAFortran)
- WRF Single Moment 6-class (WSM6) Microphysics (Fortran => C => CUDA/C)
- RRTM_LW (CUDAFortran)

Production on GPU systems: premature

Some results are reported in Figure 23-24.

WSM6: i3550 core vs. GeForce 605

Date	Step	Time (μs)		Speedup
		CPU	CPU/GPU	
20 Jan 2009	1	1761146	12496	140.93
	2	1906190	12538	152.03
	3	1929086	12526	154.00
	4	1939530	12496	155.20
	5	1952168	12448	156.82
	6	1962556	12463	157.46
	7	1967082	12443	158.09
	8	1969921	12453	158.18
	9	1971849	12517	157.53
	10	1973151	12434	158.69
10 Jul 2009	1	1738449	12371	140.52
	2	1870006	12403	150.76
	3	1889058	12411	152.20
	4	1897988	12446	152.49
	5	1906384	12390	153.86
	6	1919082	12416	154.56
	7	1926775	12420	155.13
	8	1935506	12461	155.32
	9	1941633	12475	155.63
	10	1949073	12486	156.10

Figure 23 – From: Xiao, et al., Comp & Geosci 59 (2013) 156-162

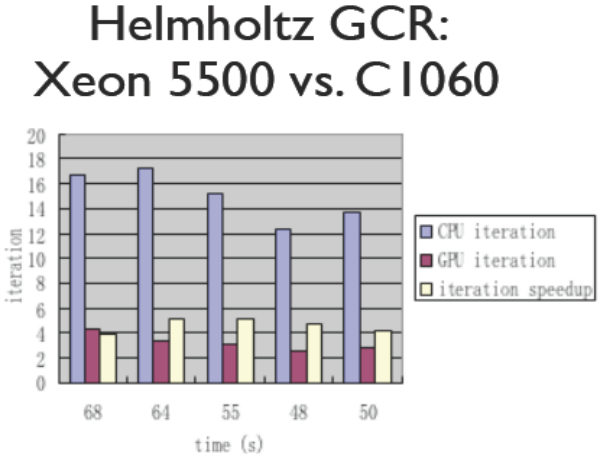


Table1. Results of GPU acceleration VS Results of CPU

module	CPU (s)	GPU (s)
helmholts	Rapid change	Rapid change
microphysics	58.54	36.10
RRTM	805	609

Figure 24 – From: Wang, et al., IEEE ICHPCC, 2011

h) COSMO

Model type: mesoscale, Cartesian Grid, finite difference, single and double precision
Paradigm: MPI + OpenACC and CUDA/C++ (through STELLA domain-specific embedded language)
Status: dynamics + physics + I/O validated. Accepted by COSMO consortium: code will come in to COSMO trunk
Some results are reported in Figure 25.

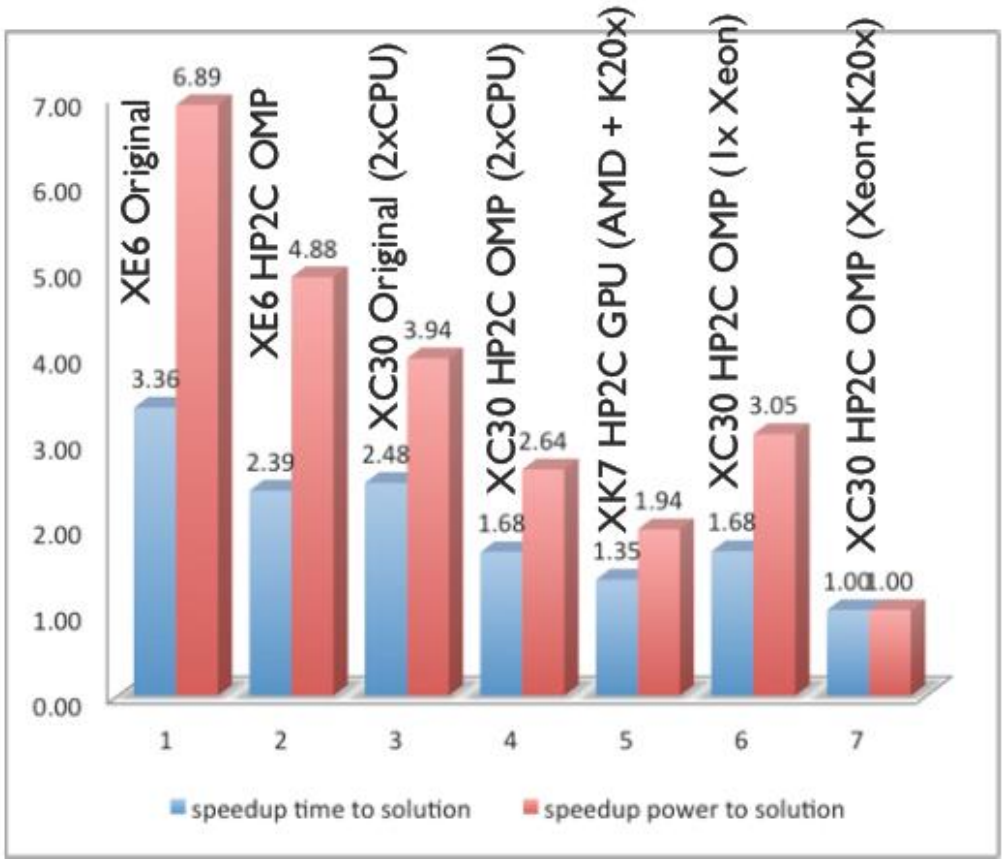


Figure 25 - Time- and energy-to-solution of original code, upgraded HP2C code on CPU (OMP) and GPU

The Intel Xeon Phi coprocessor based on MIC architecture and its direct competitor, GPU accelerators, have become widespread in scientific high-performance computing (HPC). Motivated by the increasing amount of such computing resources which are getting available to the scientific community, several HPC codes developed or used in the Max Planck Society were assessed concerning their portability and performance on compute clusters accelerated by Xeon Phis or Nvidia GPUs.
Some practical experiences and insights gained at the RZG and the DKRZ and confronts the Xeon Phi with the GPU can be taken in consideration, i.e. the VERTEX radiation hydrodynamics code, written in FORTRAN with hybrid MPI/OpenMP parallelization, in

production and under continuous development since 2001. It has been ported to all major HPC architectures (CPU, Tier-0 class) and scales up to O(100k) cores (0.25 PFlop/s sustained on 131000 core SuperMUC@LRZ), as shown in Figure 26.

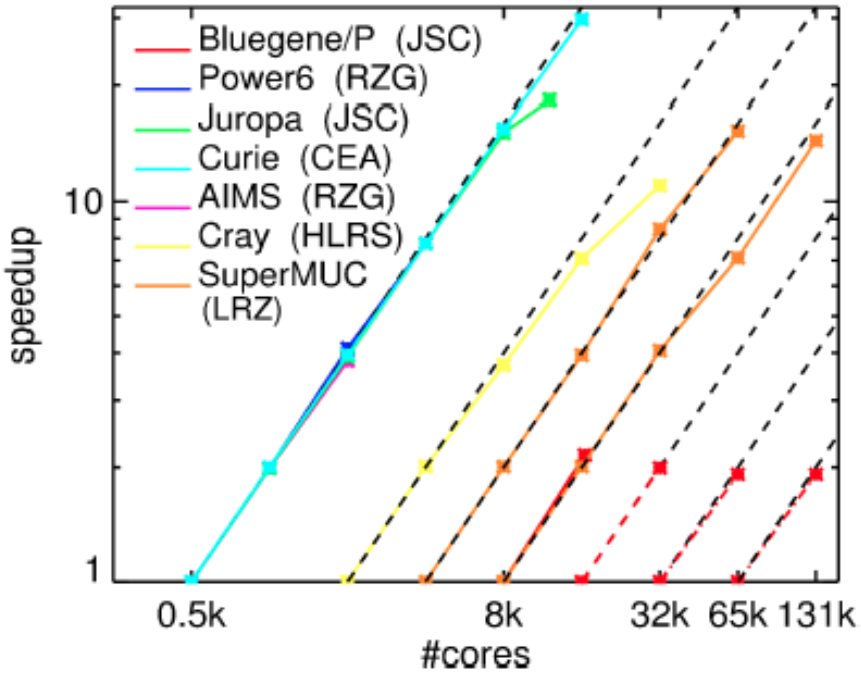


Figure 26 – VERTEX performance on main HPC architectures

Test report 2x application speedup (strong scaling) on GPU and speedup 2.5x of the kernel on MIC.

For both architectures, significant human efforts have to be invested for porting existing HPC codes such that reasonable performance gains can be obtained due to the additional coprocessors or accelerators. Contrary to Nvidia GPUs, only very few production-ready HPC applications seem to exist for Xeon Phi so far, which can partly be explained by its late introduction to the market, but also by apparent deficiencies in compilers, libraries, and the OpenMP runtime.

Nevertheless, the current generation of Xeon Phi ("Knights Corner") appears as a well-suited platform for preparing HPC codes for future manycore (e.g. the Xeon Phi "Knights Landing") and multicore (e.g. the next generation of Intel Xeon CPUs) processors.

2.4 The parareal in time method: a further direction for parallelism

The need for faster numerical simulations of complex phenomena and the definition in this context of what a complex phenomenon is, is evolving in line with the improvement of the platforms that are available for High Performance Computing. Indeed, what used to require hours or days of numerical simulations on large computers can now be run in fractions of seconds on laptops. Nevertheless the understanding of real phenomena, the control and optimization of processes and the monitoring of industrial problems propose new challenges

where i) better accuracy, ii) use of more involved mathematical models, iii) simulations on bigger object or iv) on longer period of time for unsteady phenomena are required. The evolution of the computing platforms helps in addressing bigger problems but is not sufficient.
Exascale systems, which are achievable in the next 5-10 years, will contain millions of cores. In order to make efficient use of these systems, high-performance applications must have sufficient parallelism to support parallel execution across millions of threads of execution. The development of more efficient and more highly parallel scalable solvers is therefore at the forefront of Exascale applications research and development, in particular, the domain decomposition methods or task partitioning approaches reach their limits in their ability to use the entire computational resource with the same efficiency as currently achieved on existing smaller systems.
Most simulations, which are expected to deliver scientific impact from Exascale systems, contain time-stepping in some form and present-day codes make little or no use of parallelism in the time domain; time stepping is currently treated as a serial process.
For time dependent problems, either pure differential systems or coupled with partial differential equations, the time direction leads to new families of algorithms that might allow providing full efficiencies and speed ups. The parareal (parallel in time) algorithm and the waveform relaxation methods have been introduced to fill this gap and have the potential to extract very large additional parallelism from a wide range of time-stepping application codes. This is a disruptive technology, which will deliver performance speed-ups of between 10 and 100. By comparison, optimizations of current algorithms typically yield benefits in the range of tens of percent.

2.5 Communication-avoiding algorithms

The cost of moving data in an algorithm can surpass by several orders of magnitude the cost of performing arithmetic, and this gap has been steadily and exponentially growing over time. The communication problem needs to be addressed by the numerical software community directly at the mathematical formulation and the algorithmic design level. This requires a paradigm shift in the way the numerical algorithms are devised, which now need to aim at keeping the number of communication instances to a minimum, while retaining their numerical efficiency.
In the past, some works aimed at reducing communications, i.e. overlapping communication and computation or storing redundantly data from neighboring processors for future computations (ghosting).
Recently, communication-avoiding algorithms provide such a novel perspective on designing algorithms that provably minimize communication in numerical linear algebra.
Some examples can be the parallel and sequential dense QR factorization algorithms that are optimized to avoid communication. Communication includes both messages between processors (in the parallel case), and data movement between slow and fast memory (in either the sequential or parallel cases).

Demmel et al. propose two algorithms. The first algorithm, named Tall Skinny QR (TSQR), factors m × n matrices in a one-dimensional (1-D) block cyclic row layout, storing the Q factor (if desired) implicitly as a tree of blocks of Householder reflectors. TSQR is optimized for matrices with many more rows than columns. In the parallel case, TSQR requires no more than the minimum number of messages Θ(log P) between P processors. In the sequential case, TSQR transfers 2mn + o(mn) words between slow and fast memory, which is the theoretical lower bound, and performs Θ(mn/W) block reads and writes (as a function of the fast memory size W), which is within a constant factor of the theoretical lower bound. In contrast, the conventional parallel algorithm as implemented in ScaLAPACK requires Θ(nlogP) messages, a factor of n times more, and the analogous sequential algorithm transfers Θ(mn2) words between slow and fast memory, also a factor of n times more. TSQR only uses orthogonal transforms, so it is just as stable as standard Householder QR. Both parallel and sequential performance results show that TSQR outperforms competing methods.
A second algorithm proposed by Demmel, is the CAQR (Communication-Avoiding QR), factors general rectangular matrices distributed in a two-dimensional block cyclic layout. It invokes TSQR for each block column factorization, which both remove a latency bottleneck in ScaLAPACK's current parallel approach, and both bandwidth and latency bottlenecks in ScaLAPACK's out-of-core QR factorization. CAQR achieves modeled speedups of 9.7× on an IBM POWER5 cluster, 22.9× on a petascale machine, and 11× on the Grid.

3. Co-design strategy

For the past thirty years, the need for ever greater supercomputer performance has driven the development of many computing technologies which have subsequently been exploited in the mass market. Delivering an exaflop (or 10^18 calculations per second) by the end of this decade is the challenge that the supercomputing community worldwide has set itself and requires a co-design effort in order to be efficiently exploited by the new codes. Some vendors are working on new generation architectures and measurements show that this system can make a difference for researchers because of its superior memory bandwidth per core and the resulting real application performance per core. This is in line with the general strategy, which is focusing on sustained performance, not peak or LINPACK. Many projects aim at establishing heterogeneous team to meet this issue.

3.1 CRESTA project

The Collaborative Research into Exascale Systemware, Tools and Applications project (CRESTA) brings together four of Europe's leading supercomputing centres, with one of the world's major equipment vendors, two of Europe's leading programming tools providers and six application and problem owners to explore how the exaflop challenge can be met.
CRESTA focuses on the use of six applications with exascale potential and uses them as co-design vehicles to develop: the development environment, algorithms and libraries, user tools, and the underpinning and cross-cutting technologies required to support the execution of applications at the exascale. The applications represented in CRESTA have been chosen as a representative sample from across the supercomputing domain including: biomolecular systems, fusion energy, the virtual physiological human, numerical weather prediction and engineering.
In particular, CRESTA works on the optimization of the IFS weather model, developed at the ECMWF. The IFS resolution halves about every eight years, so in 2030 it will be about 2,5km (Figure 27). Some optimization actions will be needed to sustain this resolution.

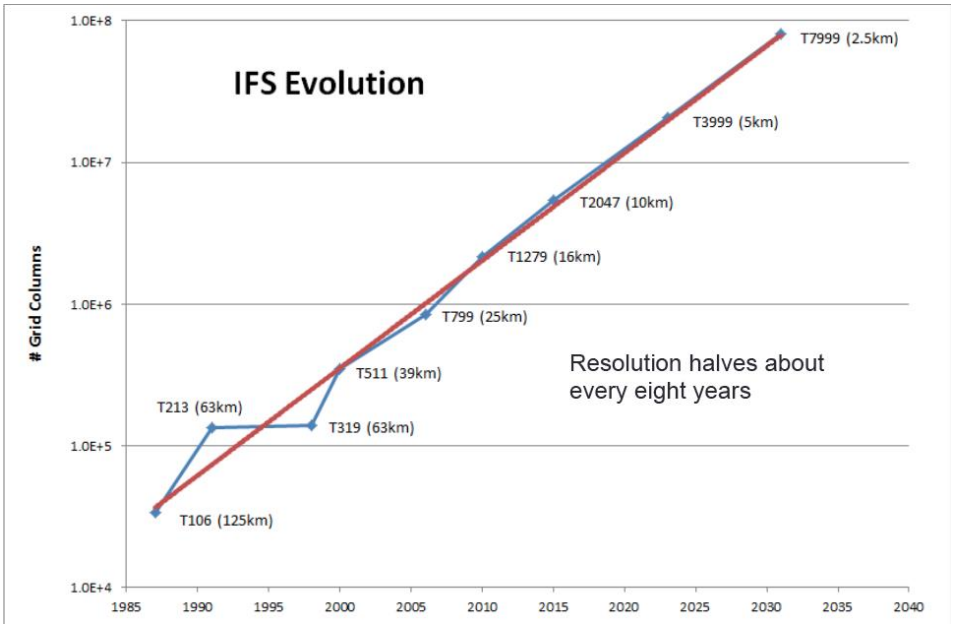


Figure 27 – IFS model resolution trend

IFS scalability optimizations in the CRESTA project regard:

- Investigation on initialization cost and propose solution.
- Development & testing of alternative local data structures (minimizing communications).
- Explore use of DAG parallelization (with OMPSs), using a toy code representative of IFS.
- Implement Coarray teams when new F2013 compiler available.

3.2 DEEP project

In the Dynamical Exascale Entry Platform (DEEP) project 16 industrial and academic partners from 8 countries develop a novel supercomputer architecture coupling a 128-node InfiniBand-connected Intel Xeon Cluster and a 512-node Intel Xeon Phi Booster on an EXTOLL 3D-torus network. This design assigns high-performance processor cores to program parts with medium scalability running on the cluster while providing highly-scalable code parts with a highly-parallel environment in the Booster.
At the Cyprus Institute the global climate model ECHAM/MESSy Atmospheric Chemistry (EMAC) is used to study climate change and air quality scenarios with a focus on the Eastern Mediterranean and the Middle East. Using the OmpSs programming model EMAC has been ported to the DEEP architecture running the spectral core and the fast physical computations on the cluster while offloading the demanding chemistry calculations to the Booster.
This approach provides an additional run-time degree of freedom in the allocation of computing resources. It allows the user to increase the number of parallel threads for high performance in local calculations while decreasing the number of tasks for spectral calculations and consequently improving implicit load balancing and reducing the amount of communication.

3.3 MONT-BLANC project

Energy efficiency is already a primary concern for the design of any computer system and it is unanimously recognized that future exascale systems will be strongly constrained by their power consumption.
Mont-Blanc is a European initiative to build prototype exascale systems using high volume commodity energy-efficient parts from the embedded and mobile markets. The project is coordinated by the Barcelona Supercomputing Center (BSC), and it started in October 2011.
The Mont-Blanc project strategy for addressing the energy consumption challenge includes the definition of an HPC System software stack (Figure 28) on ARM composed by: (i) an open source system software stack (Ubuntu Linux OS, GNU compilers, scientific libraries, slurm cluster management), (ii) runtime libraries (MPICH2, OpenMPI, Nanos++ (OmpSs)), (iii) performance analysis tools (Paraver, Scalasca), and (iv) Allinea DDT 3.1 debugger. OmpSs provides a straightforward task-parallel programming interface together with an advanced runtime system, which automatically manages load balance and data locality, and overlaps computation and communication.

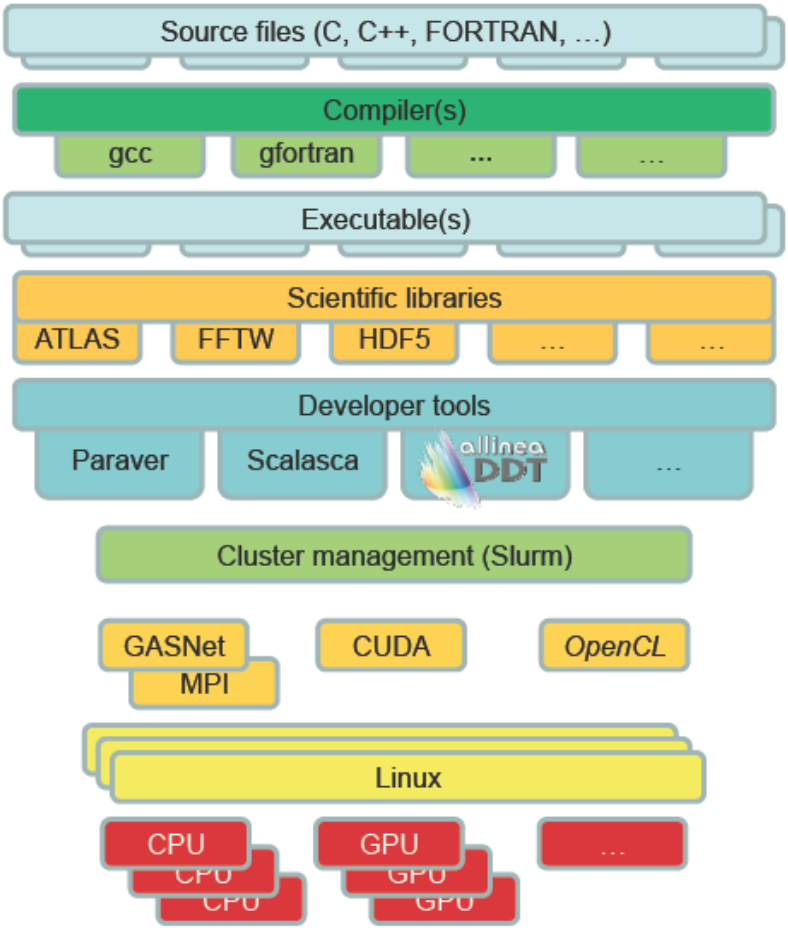


Figure 28 - HPC System software stack on ARM

Moreover, a set of micro-kernels, common to several HPC applications, has been ported to serial, pthreads, OpenMP, OmpSs, CUDA, OpenCL. Some examples are: Dense Matrix-Matrix Multiplication, 3D stencil, Sparse Vector-Matrix Multiplication.

3.4 ECMWF Scalability Programme

A Scalability Programme has been initiated at ECMWF with the aim of coordinating developments towards a more scalable Integrated Forecasting System (IFS) and the ancillary support software. The Programme will provide accurate, efficient and scalable algorithms and code structures to cater for a variety of potential future high-performance computer (HPC) architectures.

ECMWF Scalability Programme objectives are to:

1. Develop the future IFS combining a flexible framework for scientific choices to be made with maximum achievable parallelism.
2. Prepare for expected future technologies and their implications on code structure ensuring efficiency and code readability.
3. Develop environment/metrics for quantitative scalability assessment, through
   - coordinating ECMWF-internal resources, R&D strategy,

- engaging with external partners (Member States, academia, HPC centres, vendors).

ECMWF is not alone with this concern and worldwide efforts are targeted towards improving the scalability of operational large-scale software applications. The Programme will coordinate resources from across the Centre to define the future forecasting system across all scales. The projects that form parts of the Scalability Programme will be run under the formal project management methodology. This encompasses the management, control and organisation of a project and enables the successful delivery of projects within time, cost and quality constraints.
Collaboration is crucial for the Programme's success. In addition to Member States, some partnership with consortia like HIRLAM, ALADIN, COSMO, NEMO, NEMO-VAR, HPC centres, the climate modelling community and hardware companies, have been estabilished. Vendors have an important role in the design and providing advice and access to the latest computing architectures. The collaboration with HPC centres allows to run the development codes on emerging novel computer hardware so that we can make informed decisions on their efficiency.
Two different approaches have been considered:

- In the short-term some actions, such as to overlap communication/computation, to outsource (e.g. radiation on GPUs), to extend co-array structures, to employ OpenMP4 once available, to use OpenACC, to migrate towards a single executable, to distribute I/O.
- In the long-term other actions, such as alternatives to global spectral model, local data structures, more flexible top-level control structures, sub-windows in sequential data assimilation.

4. Future Coupling Technologies – towards interoperability

4.1 Motivation

The flexible deployment of Earth System Models requires not only that they be loosely-coupled to their peer models, but also that the code comprising these models is loosely-coupled to the coupling technology and other communication libraries it employs. To facilitate this objective, this section describes an object-oriented framework, which provides communication functionality primitives (for example, supporting halo exchange and coupling data exchange) to client model code in a manner, which hides the particular underlying technology in use. This approach permits models to maintain a consistent interface to multiple coupling and communication toolkits, such as MCT and ESMF, and allows the interoperable use of either or both of these libraries within the models of a coupled model with only trivial configuration changes.

This work builds on work undertaken in the original IS-ENES project (see deliverable, D8.5 from IS-ENES, "Towards flexible construction of ESMs using BFG") and, in IS-ENES2, relates to activity in WP10/JRA2 as well as WP3/NA2.

4.2 Overview

Figure 29 illustrates the design of the class hierarchy which implements the interoperable framework developed in this work. This has been coded in Fortran 2003, and offers an extensive API to client models which represents an abstraction of the operations provided by the underlying communications technology.

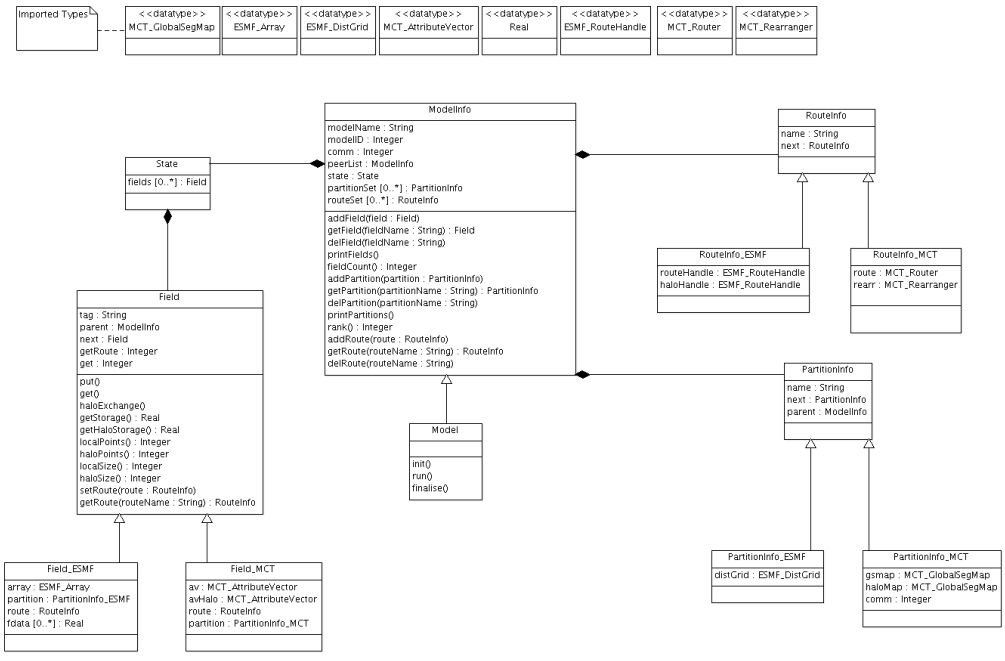


Figure 29 - Framework Type Hierarchy

To demonstrate the operation of this framework, two simple benchmark models have been developed which use the framework to perform their coupling and halo-exchange routines. Each of these models subclasses the API's ModelInfo class resulting in a simple and intuitive code structure, yet which has access to the functionality and state-management routines offered by the framework.

One of the characteristics of this implementation is a separation between the specification of the Fields that have existed in a model and their subsequent use in the computational kernel of that model. This permits the particular coupling and halo-exchange facilities used by a model to be changed with only minimal alteration to the source code. As Figure 30 depicts, this technique also permits the framework to allow the use of a "Configuration Module" in which each Field can be configured to use either ESMF or MCT as its underlying communications technology by specifying this as settings in an external metadata file that is read in by the main program in a preparatory stage.

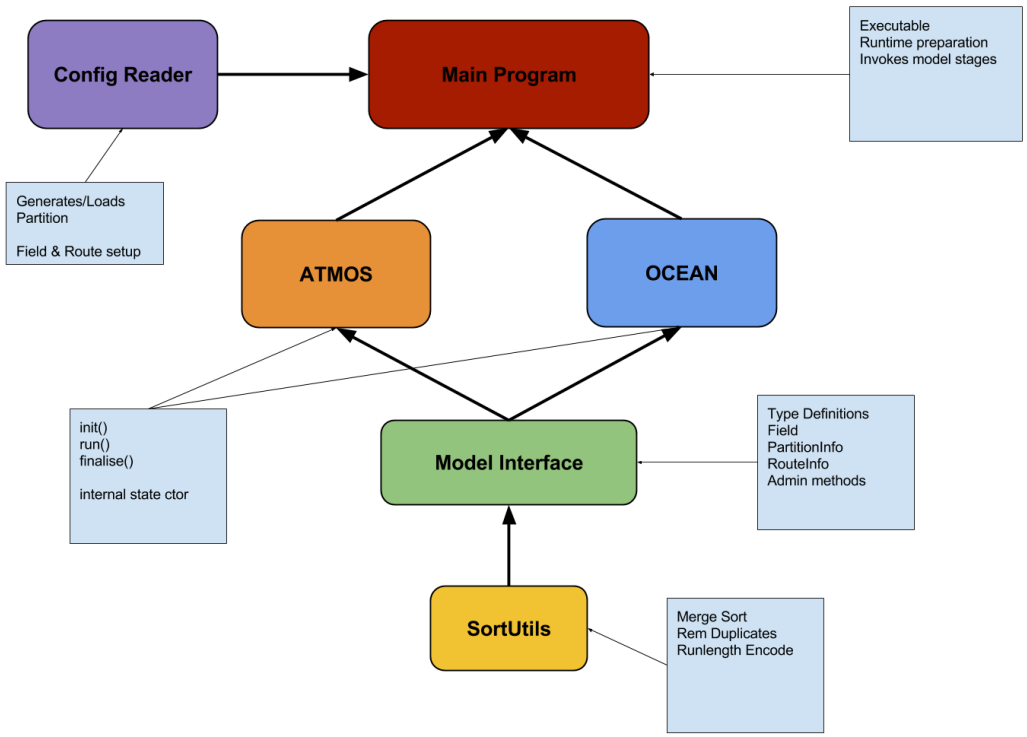


Figure 30 – API structure illustrating external configuration

An additional benefit of this modular structure is the ability to compile model code separately from the code implementing the interoperable framework. This greatly simplifies the task of

distributing both models and framework and of porting their code to new execution environments.

4.3 Implementation and the Role of Abstraction

The ModelInfo type is central to the framework's abstract representation of a scientific model, and aggregates within itself a number of entities which exist within each model that is an instance of it. These entities collectively represent the state of the model and specify which particular underlying communications technology will be used for each entity. The three most significant of these entities, which characterize a model's content and connectivity are:

1. The Field; a container object in which a model's data is stored, and which acts as the basic unit of communication both within and between models.
2. The PartitionInfo; a description of the decomposition of the model's input data and its distribution between the PEs executing the model code.
3. The RouteInfo; a description of the connectivity between models, and between regions of the partitioned model data in an individual model.

The ModelInfo class contains an interface for the registration, indexing, and deletion of these entities (using the various add(), get() and del() methods) which provides client models with a consistent and implementation-independent mechanism for specifying their configuration. This mechanism has been explicitly designed to simplify the code of client models and to enable the configuration of models using code, which is automatically generated from metadata, perhaps by a coupling framework generator such as the Bespoke Framework Generator, BFG (see D8.5 from IS-ENES for more information).

Within an instance of the ModelInfo class, each of these entities is considered as an abstract entity representing the essential characteristics of its type. The detail of the specific concrete type created is determined by an initialization routine within the model code itself. This is achieved by coding the ModelInfo to deal with abstract base classes of each Field, PartitionInfo or RouteInfo, for each of which there exists a set of concrete subclasses, with one concrete type for each of the underlying communications technologies which can be used for that purpose. For example, the abstract Field type has subclasses of both Field_ESMF and Field_MCT, both of which have very different internal structures according to the needs of their respective technologies, but which both maintain the same interface of public methods, so that client model code may treat any field as being of a generic type, irrespective of the actual form used. In the model code itself, a pointer to the abstract base class may be used to invoke any of the methods supported by the concrete types, without requiring the model to know which type it is actually operating upon.

It is primarily through this means that a high level of interoperability has been achieved using this framework. Instances of the Field, PartitionInfo and RouteInfo types, each of which might employ different underlying communications technologies, may be used simultaneously, interoperating with entities which use a different technology. In addition, this

technique permits the framework to be extended to support additional communications technologies by creating further subclasses for concrete types, which implement the abstract base class's interface in a manner specific to the library that they represent.

4.4 Performance and Extension

In addition to the functionality offered by this approach, in terms of the range of communication operations available to client models and the ease with which they may be incorporated into model code, the framework described above also offers a high level of performance, in that it adds a very minimal overhead to any client models which employ it. As implemented in Fortran 2003, the framework compiles to native code in which the only overhead attributable to its object-oriented design is the indirection that results from resolving virtual methods in abstract classes to their appropriate implementations at runtime. Significantly, the design of the framework incorporates a getStorage() method in the base Field class, which provides each model with a pointer to its local data. This permits the most computationally intensive operations constituting the numerical kernels of the model to be performed directly on data in local storage and without any intermediation or indirection on the part of the framework.

As currently implemented and described in Figure 30, the framework uses only simple models which each operate upon different partitions of a single mesh. While this is adequate to demonstrate the correctness, ease of coding, and numerical performance of the framework, many real-world models incorporate operations such as regridding to translate their data-space into a form suitable for use by a peer model. In the circumstance that a model developer wishes to implement a regridding routine, or indeed any other application-specific operations that are not directly provided by the framework, the framework's hierarchical structure permits it to be considered as a prototype API which can be subclassed by the model developer in order to provide additional functionality specific to that model, yet which maintains the capability to interoperate with the existing classes and ModelInfo type. As the ModelInfo interacts with its aggregated Field, PartitionInfo and RouteInfo instances by way of base class pointers, an appropriate derived type, which conforms to the required abstract virtual interface, automatically inherits its essential functionality and need only have the application-specific code added.

5. Conclusions and Recommendations

The next generation exascale systems will help climate science due to the availability of a huge computational power. The resource capacity of the emerging architectures will be easily exploited by increasing the ensemble size/duration or the complexity of the Earth System Models. Meanwhile, the exploitation of the resource capability to increase the models resolution will require a deep re-visiting of the models themselves to improve their scalability. To this end, some recommendations can be provided as outcomes of the analysis provided by the present deliverable: the need (i) to integrate into legacy codes new parallel models and algorithms, (ii) to tackle the I/O bottleneck to scalability, (iii) to take into consideration a co-design approach for the hardware and software development and (iv) to adopt new coupling strategies.

Today, many climate models are parallelized using the Message Passing paradigm. When the number of parallel processes increases up to several thousands of cores, some of the main bottlenecks to scalability are the communication and memory access overhead, such as the workload imbalance among the parallel processes. The introduction of a hybrid programming model, based on the use of the message passing coupled with a shared memory paradigm, allows the reduction of communication, improving the cache memory locality (due to the cached data sharing) and the load balance (due to a more dynamic allocation of tasks/data to the available resources). The hybrid programming model can be fully exploited on emerging hybrid architectures, which are very often equipped with co-processors (e.g. the Intel Xeon Phi) and accelerators (e.g. GPGPUs). One of the main recommendations to the climate science community is the need to take into consideration the new trends in parallel programming for the development and the improvement of both new and legacy codes. The porting is not easy and requires the effort of large development teams. For example, few models are currently executed on Xeon Phi co-processors due to its only recent introduction to the market and some inefficiency of compilers and libraries, etc. The porting of hybrid parallel (i.e. MPI/OpenMP) applications seems to be rapid (sometimes only requiring re-compilation), but an efficient use of the new hardware resources requires a deep re-visiting of the code. Meanwhile, the porting of the legacy codes to GPGPUs requires the adoption of an "all-or-nothing" strategy, since the kernels of the same time loop have to be executed on the same hardware resource (GPGPU or CPU) in order to limit the data traffic on the bus. However, this strategy is in sharp contrast to the continuous need to improve the model's scientific performance concurrently with the real time execution performance.

Currently, climate models are parallelized on the spatial domain. The increase of spatial resolution implies the increase of time resolution and the sequential nature of simulation in time direction limits scalability. The parallel-in-time (PINT) methods aim at filling this gap. However, the adoption of these methods requires a deep re-design of the solvers at numerical level. A strong investment in this direction is needed, justified by the radical improvement that they promise.

Moreover, to overcome the bottleneck arising from the communication overhead, the adoption of techniques limiting the communication's effect on the execution time, for example, by overlapping communication and computation or by reducing the communication frequency through the use of a redundant computation (ghosting technique), is not enough. A recommendation is to work on the integration of advanced numerical libraries, which allow reducing the communication overhead at numerical linear algebra level.

In many climate models, the spatial resolution improvement allowed by the computational capability of the new generation systems results in the production of increasingly huge quantities of data, so that one of the main bottlenecks to the model's scalability is a result of the I/O operations, in addition to data management issues. Often, the scalability analysis performed on climate models does not include I/O due to its massive negative impact on performance. The development of applications able to efficiently use emerging architectures has to exploit parallelism not only at computational- but also at I/O-level. In recent years, several solutions have been proposed at the different layers of the I/O stack, e.g. parallel file systems, I/O middleware, high-level I/O libraries. The recommendation to climate model developers is to integrate these solutions to limit the effect of I/O operations on model performance.

The improvement of scalability requires a co-design approach to be followed for hardware and software development. This is also a conclusion in the *Applied Mathematics Research for Exascale Computing*[2] recommendations, "an intensive co-design effort is essential for success, where computer scientists, applied mathematicians, and application scientists work closely together to produce a computational science discovery environment able to exploit the computational resources that will be available at the exascale". Starting from the analysis of new mathematical and algorithmic approaches, a set of solutions has to be identified taking into account both the users' community requirements and the new hardware trend. The main recommendation is to put together heterogeneous expertise in order to develop new hardware and software for high-performance and energy-efficient solutions.

Finally, the performance of Earth System Models is often affected by the use of inefficient coupling systems. The main recommendation is to analyze the performance characteristics of the proposed coupling infrastructures (taking into account the benchmarking results) and to evaluate the benefits deriving from the adoption of different coupling solutions. The development of interoperable frameworks, allowing the use of multiple coupling technologies in a single coupled application, helps the end-user to improve the coupling performance by limiting the integration effort required

[2] *Applied Mathematics Research for Exascale Computing*, U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research Program, 2014

To conclude, re-visiting the models requires the work of heterogeneous and complementary research teams, increasing the staff costs needed to address the issues involved. The actions required could be performed at different levels, but in many cases there will be strong dependencies between the solutions required. Hence, the development is hard to parallelize, though this is a major requirement in order to reduce the time-to-market and to develop models ready for exascale in time to exploit the capability of next generation architectures.