Report from the Crossing the Chasm Workshop

Report compiled by Mike Rezny,
Monash University, Australia
michael.rezny@monash.edu
Edited and Revised by Bryan Lawrence and Reinhard Budich

Report from the ENES Strategy Meeting
Reading, 26th October 2016
For the IS-ENES2 GA
16th January 2017

Incorporating (hacked) slides and ideas from:
Joerg Behrens,
Willem Deconinck,
Rupert Ford,
Chris Maynard,
Steve Mullerworth
Carlos Osuna,
Andy Porter,
Kim Serradell
Sophie Valcke,
Simon Wilson
…and the other attendees.

PROBLEM STATEMENT

Science Complexity is increasing:

- More components
- More complex interactions between components
- More derived data
- More complex and much larger ensembles
- More observational data for assimilation
- Higher resolution
- More complexity in the grids / meshes
  - Unstructured, extruded meshes
  - Multiple meshes
  - Multi-grid methods
  - High-order, mixed finite element methods

Boundary Conditions:

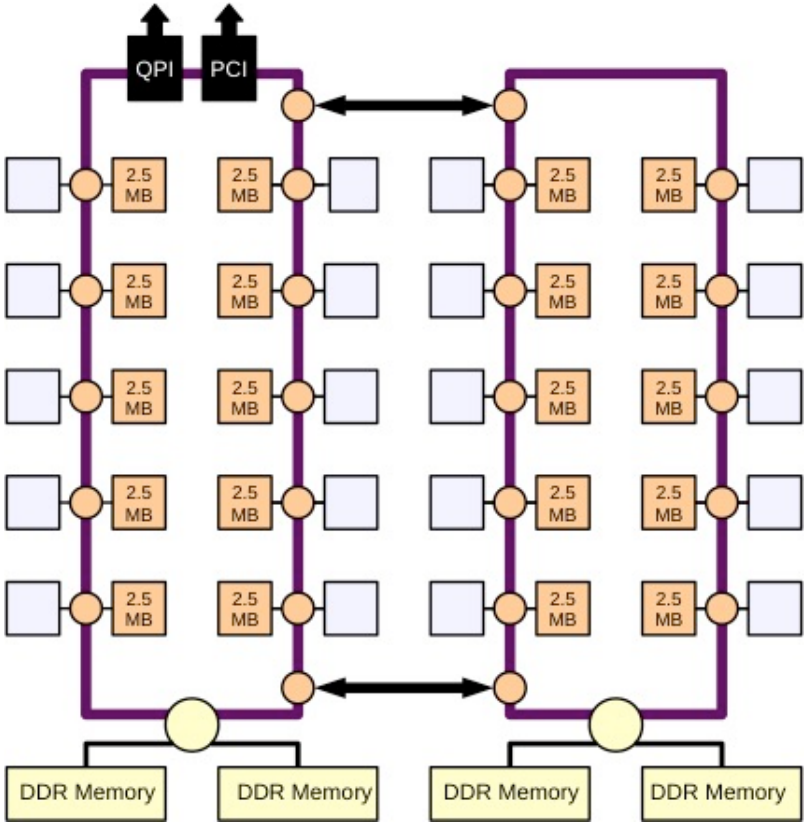
- Little or no increase in the performance of each core
- Increased performance is needed to meet research and operational requirements

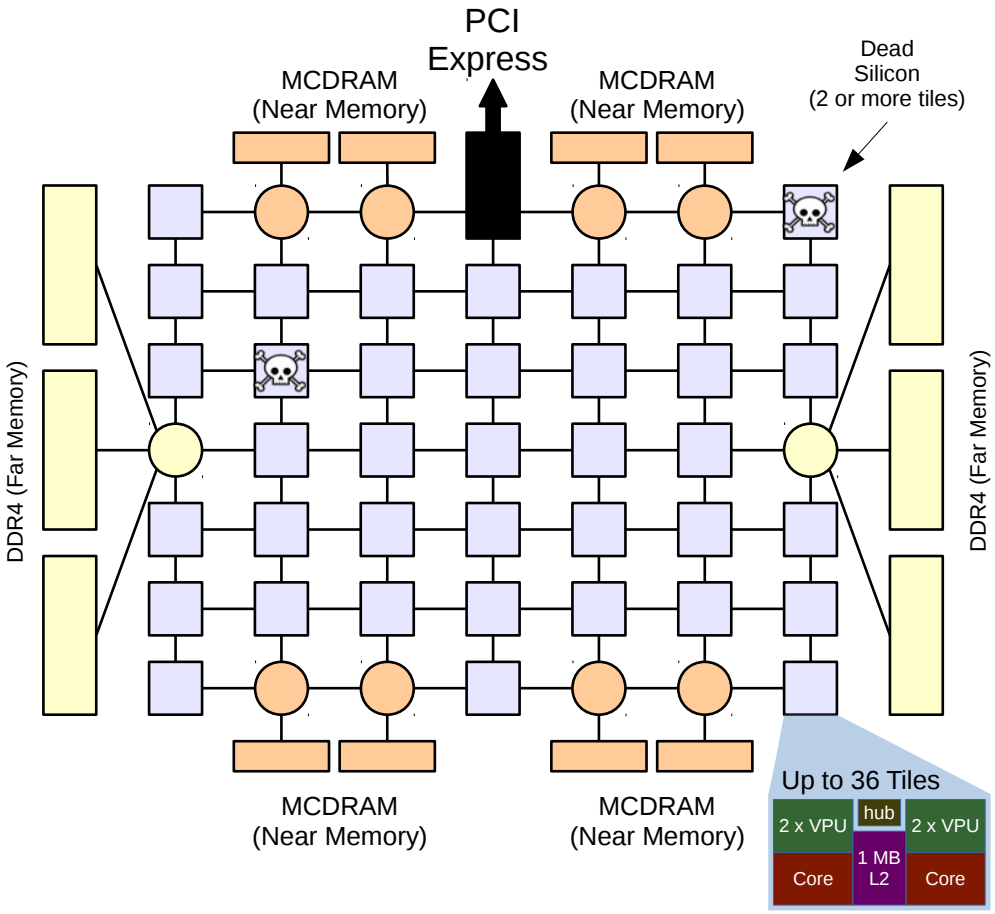
Computational Complexity is increasing:

- More cores per socket are provided
- Performance improvement is mainly by higher levels of parallelism
- Heterogeneous hardware architectures with accelerators
- Increasingly complex solutions are being applied to achieve the required performance:
    - Numerics
    - Algorithms
    - Software design
    - Workflows

The difficulty is in applying these **complex computational** solutions to existing large **complex science** applications

Broadwell and Knights Landing





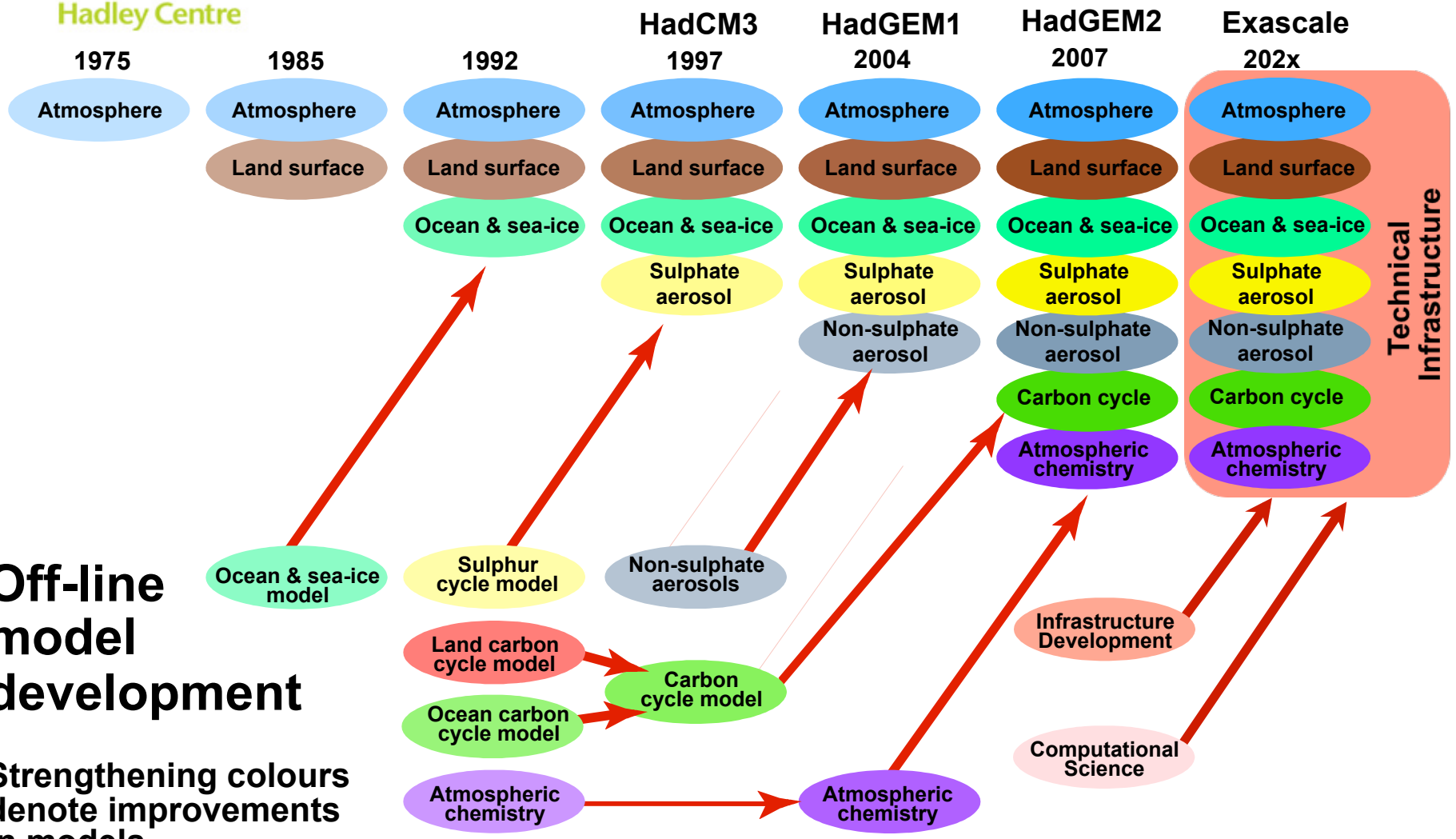
Cores aren't even the big difference! Memory, cache, and bandwidth differences

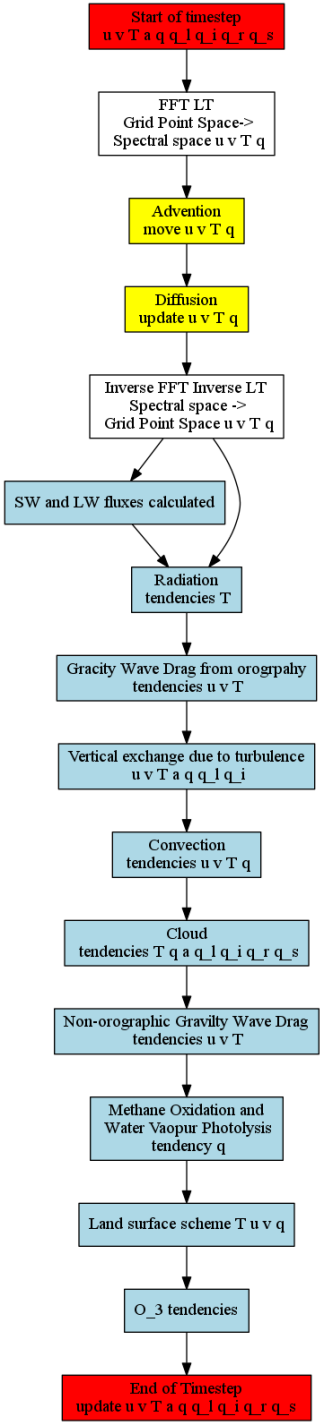
One Example of Increased Computational Complexity: **Concurrency**

Present and future models may need to incorporate up to **7** types of concurrency to meet performance requirements with anticipated hardware.

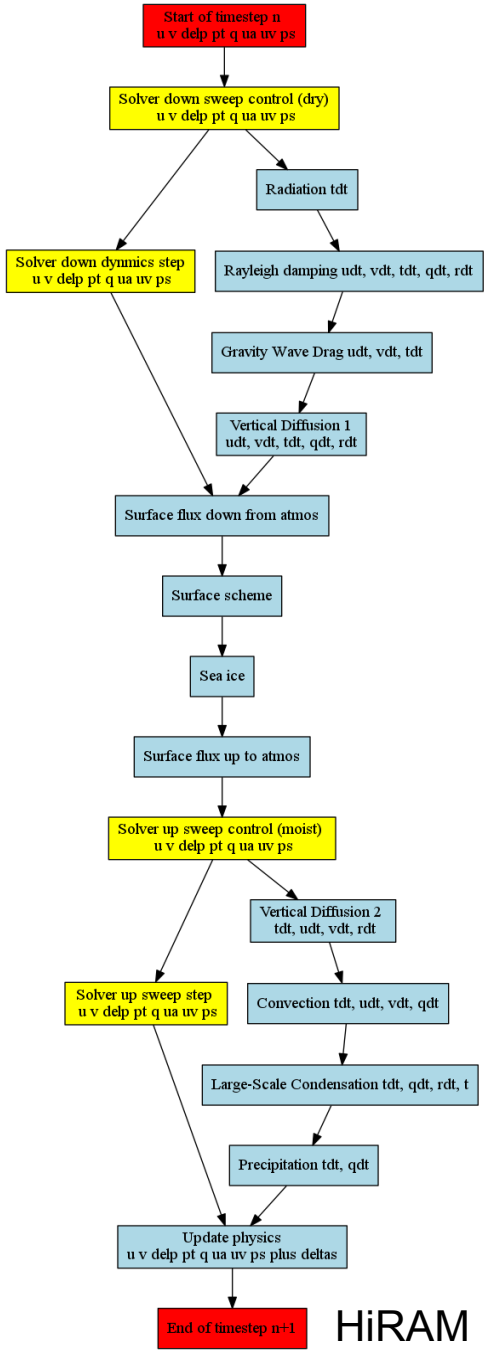
- Vectorisation
- Distributed-memory concurrency across nodes / sockets (MPI)
- Shared-memory concurrency within nodes / sockets (OpenMP)
- External coupling of independent models (Oasis-MCT, Cpl, YAC)
- Asynchronous I/O (XIOS, I/O servers)
- Executing model components concurrently (ESMF, FMS)
- Hardware accelerators (GPU, OpenACC)

Including Exascale Development Support

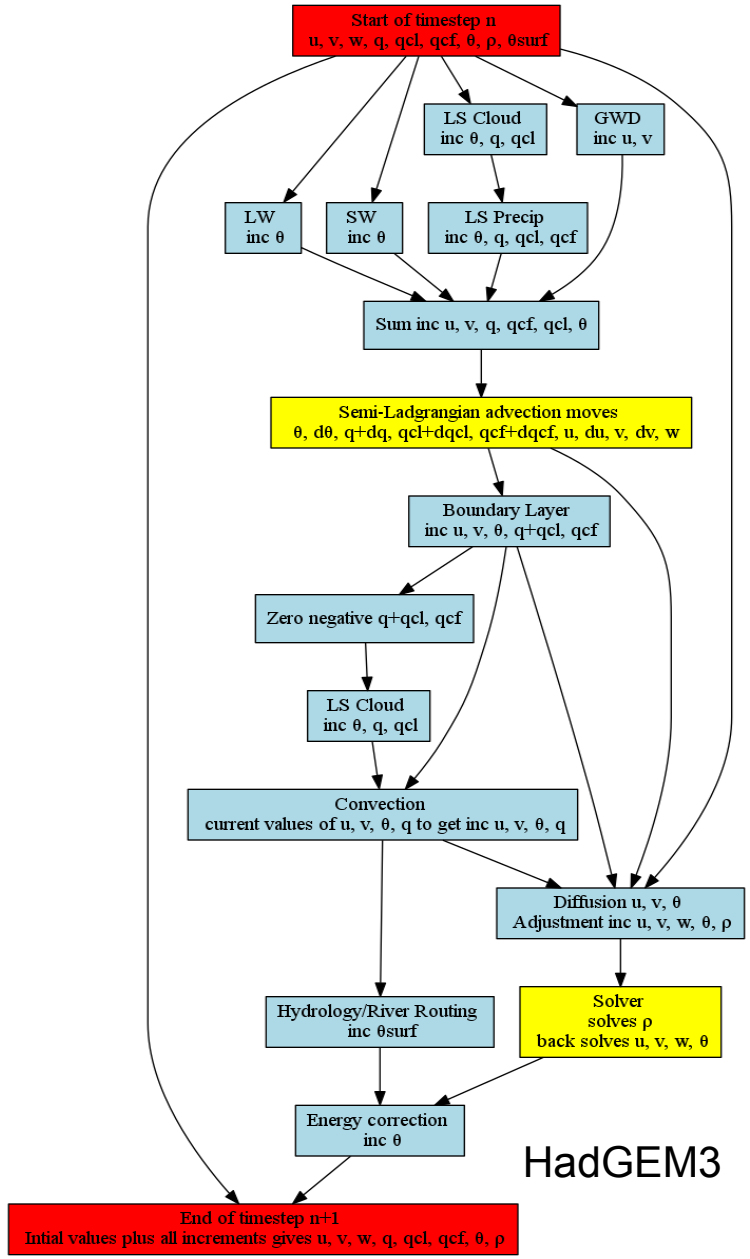




IFS

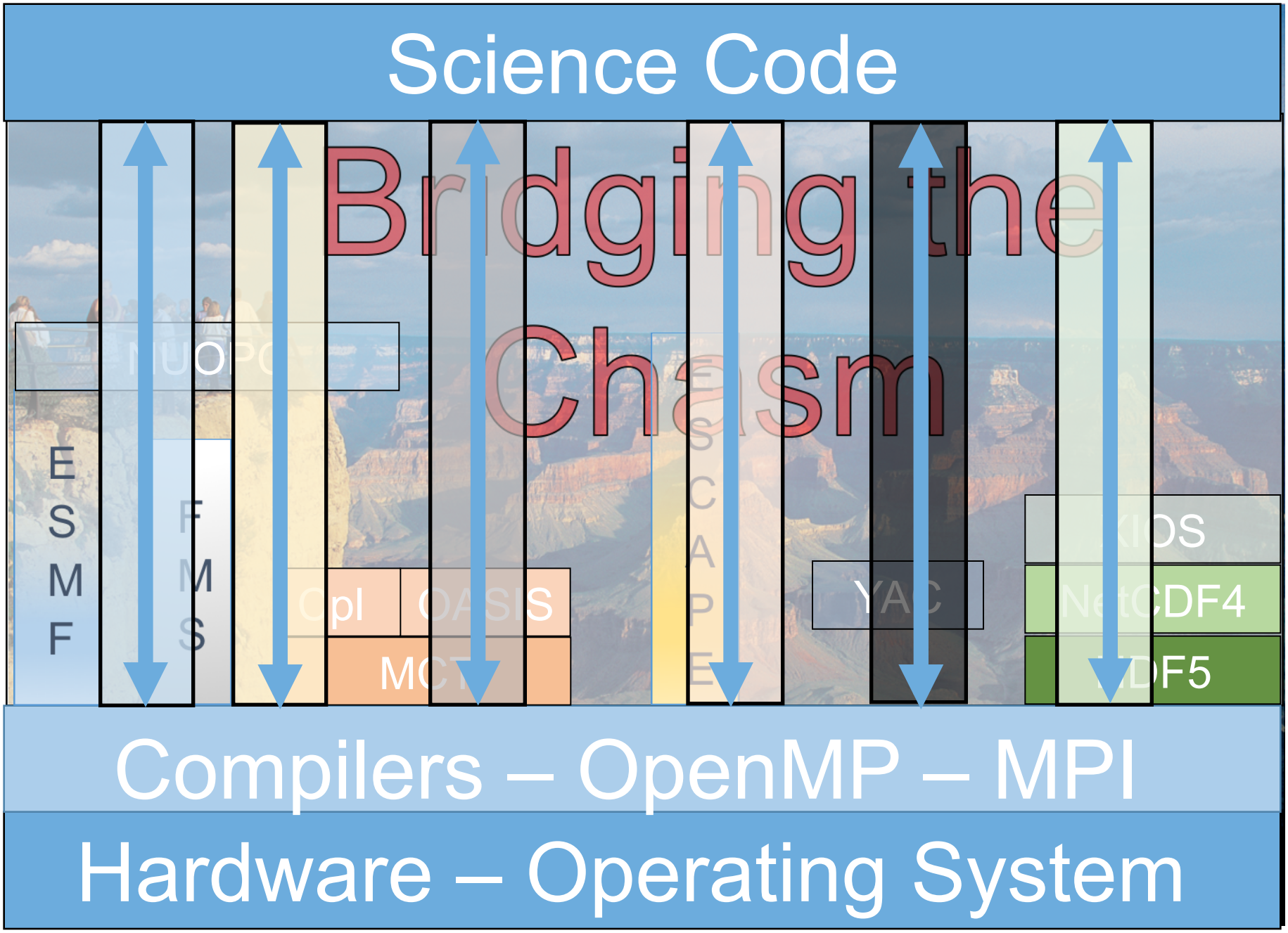


HiRAM



HadGEM3

Dynamics
Physics
Start/End
Service


Science Code
Bridging the Chasm
ESMF
FMS
Cpl
OASIS
MCT
YAC
XIOS
NetCDF4
HDF5
Compilers – OpenMP – MPI
Hardware – Operating System

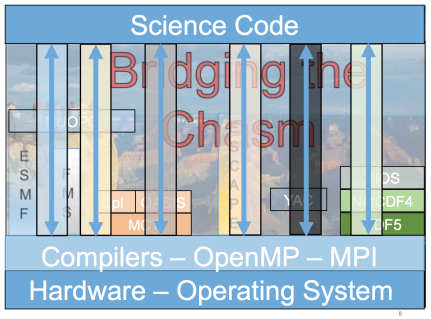
Can we, as a community, work collaboratively to:

- Discuss,
- Specify,
- Design,
- Develop,
- Maintain, and
- Document

libraries and tools that will help all of us to

Progressively Reduce the Chasm in ESM development?

From The top:



DOMAIN SPECIFIC LANGUAGES

Hartree Centre
Science & Technology Facilities Council

What is the problem? I

Codes

- Complex and evolving science
- It is hard to restructure codes
- Software has a long life (20 years +) so outlives hardware architectures

Software

- Compilers complex and evolving
- Standards evolving (OpenMP, OpenACC, MPI, …)

Hartree Centre
Science & Technology Facilities Council

Is there a solution?

Separation of concerns

- Separate science code from parallelisation and optimisation

- Single science source
- Targeted parallelisation and optimisation for portable performance

- Achievable using domain specific knowledge … Two important questions:
  - what is a domain?
  - What is the language

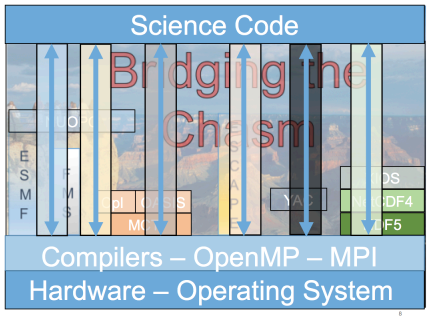
Experience thus far

1. Separation of concerns and auto-code generation have allowed LFRic to scale to large number of cores (Data Parallelism)
2. Also target shared memory parallelism via OpenMP
3. No science code was altered (or harmed)
4. Vertically contiguous data layout hides indirect memory access cost of unstructured mesh

1. Algorithmic developments (Multi-grid etc) will improve scaling (next target is 221,184 cores 1c)
2. Can switch programming model (OpenACC) for GPU
3. Also looking at Vulcan (ARM) and Knights Landing
4. Look at ILP (kernels) and Task parallelism (driver layer)

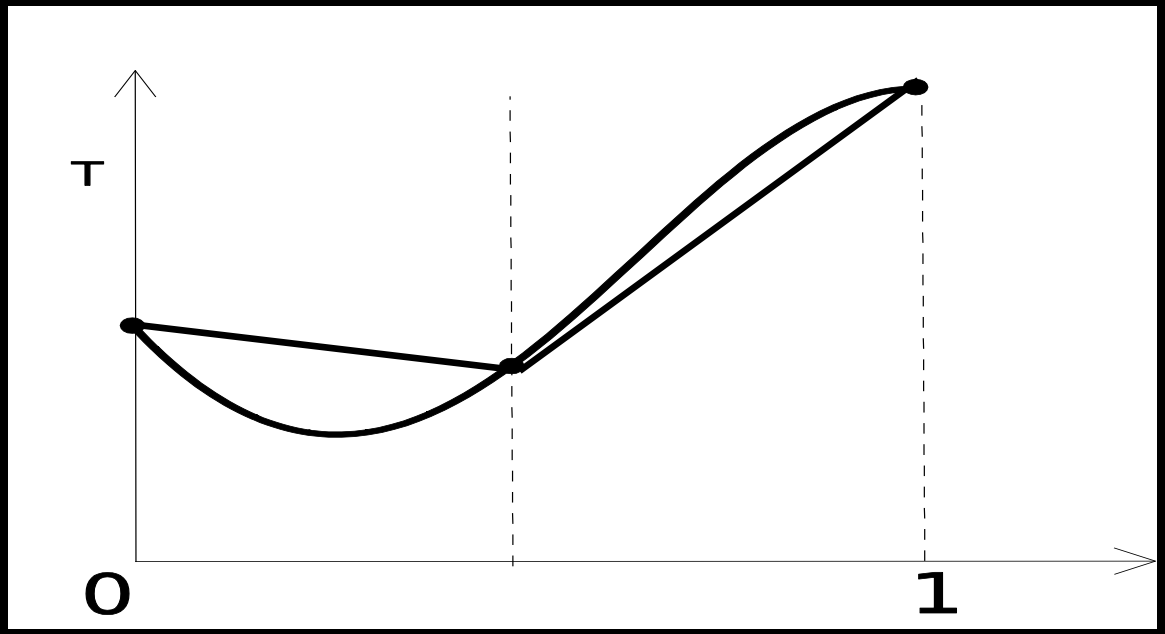
… not dissimilar results with GridTools (same idea different implementation)

From The top:



NEED FOR A DATA MODEL

What is a data model?



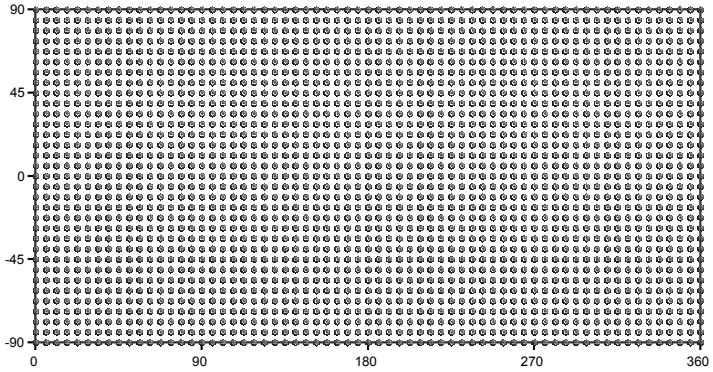
- How do we store the data?
- What happens between data points?
  - Supports science operations and coupling
  - Influences distributed and shared memory deployments

What is Atlas?

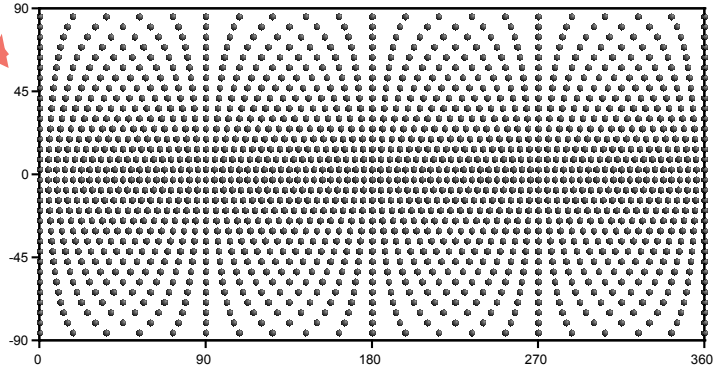
A flexible parallel framework for

- structured grids

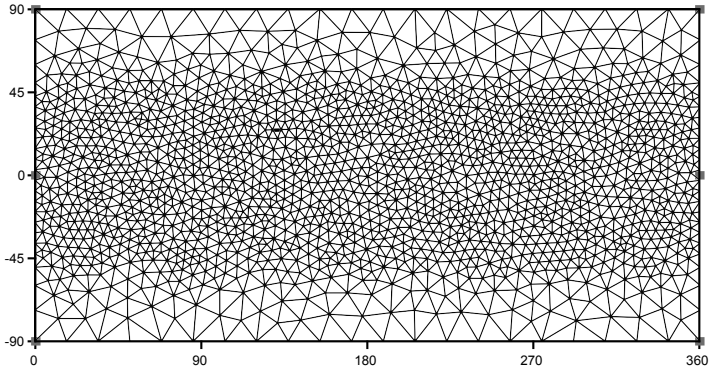
unstructured hybrid meshes



Regular Lat-Lon (L32)



Octahedral Gaussian (O32)



Unstructured mesh

- Parallel mesh generation
- Mathematical operators
- C++ / Fortran

ECMWF

Planned Atlas developments

- Overlapping communications and computations
- Data storage for many-core architectures
- Parallel interpolation operations
- Spectral/Finite Element basis functions, quadrature
- Serialisation of all data structures to memory (caching)

Met Office
Hadley Centre

Can we agree on generic data model requirement that meet the need of all ESMs?

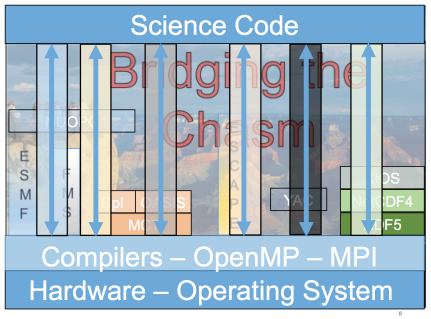
Does a one-size fits all approach support both scientific and compute performance goals?

I dunno!
Je sais pas
ich weiß nicht

Summary DSL thoughts

- To cross the chasm we should aim for
  - single source science codes (single source, multiple architectures),
  - higher level specifications and
  - performance portability
- DSEL's are a potential way forward
  - Stella/GridTools and PSyclone/LFRic provide demonstrators
  - Mostly complimentary work
  - Perhaps PSyclone could use Stella?
- Potential for collaboration with optimisations?
  - Convince other groups to try these out
  - Absolute need to involve VENDORS
- Data layout is still a potential issue
  - (data models, un-structured grids)

In the middle?:



UNDERSTAND THE BUILDING BLOCKS

Funded by the
European Union

ESCAPE key objectives

- Define fundamental **algorithm building blocks** ("*Weather & Climate Dwarfs*") to co-design, advance, benchmark and efficiently run the next generation of NWP and climate models on energy-efficient, heterogeneous HPC architectures.

- Combine frontier research on **algorithm development** and extreme-scale, high-performance computing applications with **novel hardware technology**, to create a flexible and sustainable weather and climate prediction system.

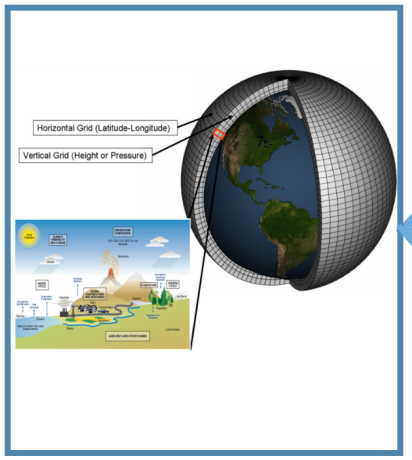
- Foster the **future design of Earth-system models** and commercialisation of weather-dependent innovative products and services in Europe through enabling open-source technology.

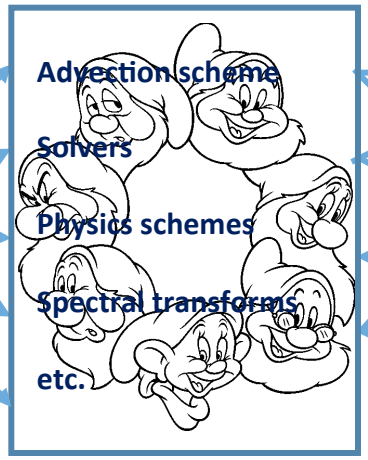
- Pairing **world-leading NWP** with **innovative HPC solutions**.

Funded by the
European Union

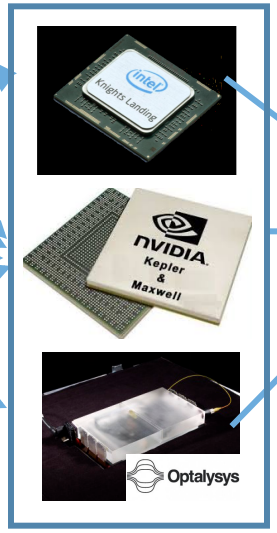
ESCAPE

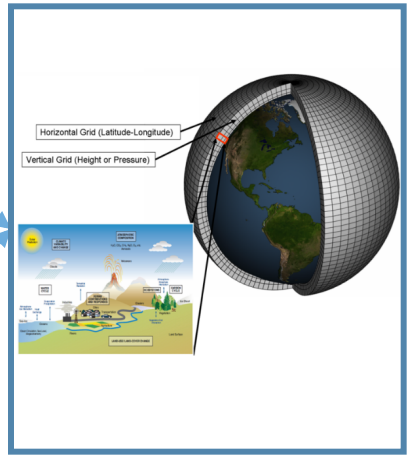
Disassemble global … NWP model → **Extract, redesign… key components** → **Optimize for energy… efficiency on new hardware** → **… Reassemble global NWP model**





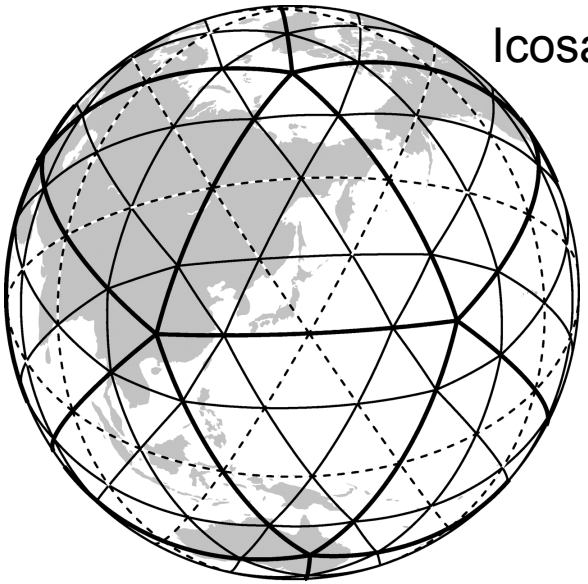
(memory, communication, compute intensity)



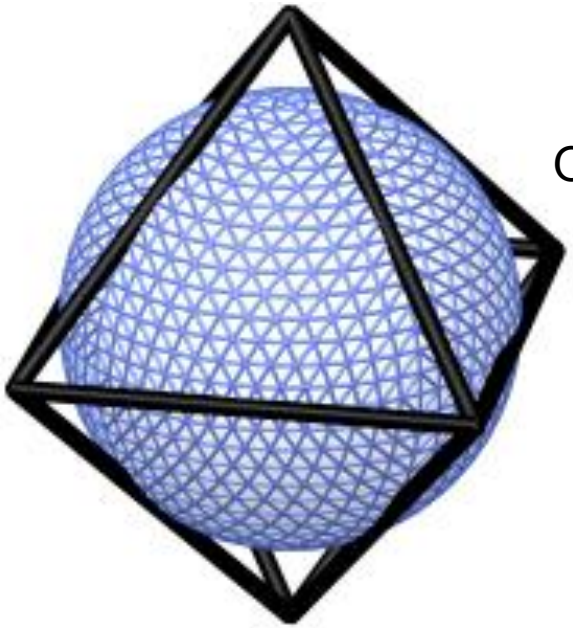


ECMWF · MeteoSwiss · DMI · RMI · NVIDIA · METEO FRANCE Toujours un temps d'avance · ICHEC · Deutscher Wetterdienst Wetter und Klima aus einer Hand · Optalysys · Bull · PSNC · Loughborough University

Funded by the
European Union



Icosahedral



Octahedral

Cube sphere

GridTools On Irregular Grids

Dual-grid

ECMWF
MeteoSwiss
DMI
RMI
NVIDIA
METEO FRANCE
Toujours un temps d'avance
ICHEC
Deutscher Wetterdienst
Wetter und Klima aus einer Hand
DWD
Optalysys
Bull
PSNC
Loughborough University

Funded by the European Union

Dynamics

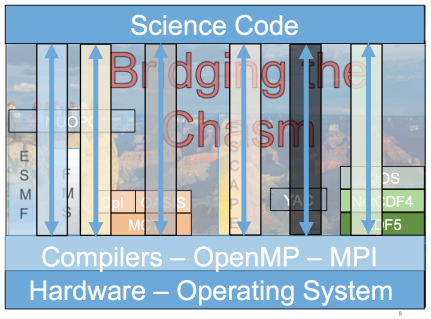
Advection (SL, MPDATA)
Elliptic solver (GCR)
Spectral transform (SH+

Physical parameterisation

Cloud Microphysics (CloudSC)
Radiation (ACRANEB2)

Dwarf		Target Accelerators			Programming Models			
	Priority	GPU	Xeon Phi	Optalysis	MPI	OpenMP	OpenACC	DSL
D - spectral transform - SH	I	✔	✔	✘	✔	✔	✎	✘
D - spectral transform - biFFT	I	✔	✔	✔	✔	✔	✎	✘
D - advection - MPDATA	I	✔	✔	✘	✔	✔	✎	✎
I - LAITRI (3d interpol. algorithm)	I	✔	✔	✘	✘	✔	✎	✎
D - elliptic solver - GCR	II	✔	✔	✘	✔	✔	✎	◌
D - advection - semi-Lagrangian	II	✔	✔	✘	✔	✔	✎	✎
P - cloud microphysics - CloudSC	II	✔	✔	✘	✘	✔	◌	✘
P - radiation scheme - ACRANEB2	II	✔	✔	✘	✘	✔	✎	✘

From the bottom:

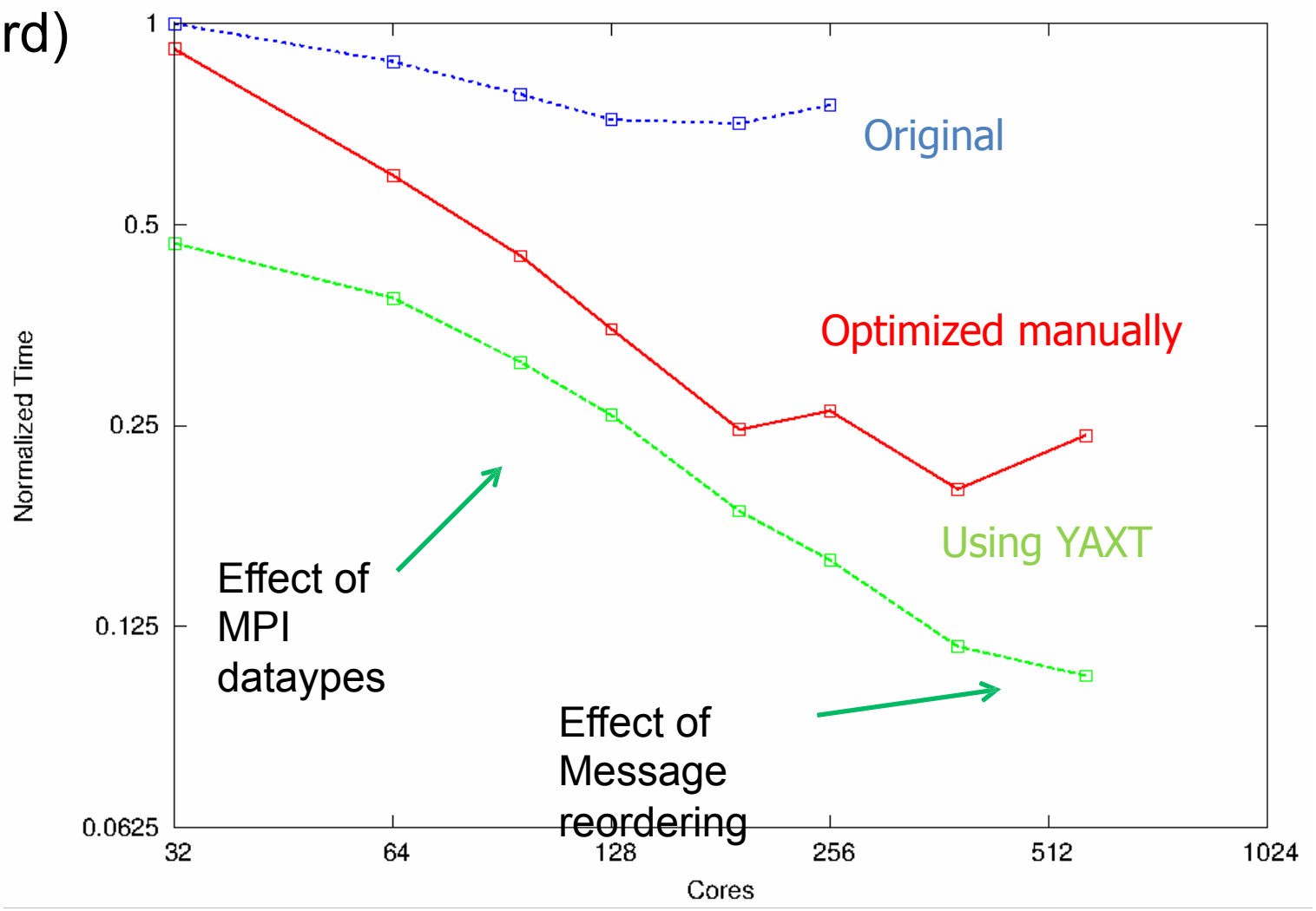


WORK SMARTER WITH MPI

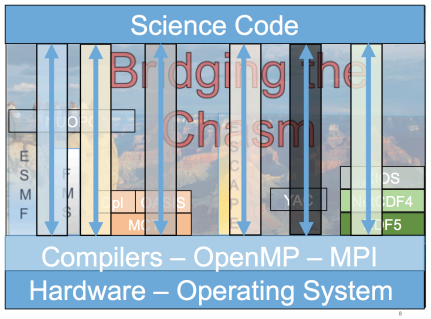
YAXT: performance – example ECHAM

Global Transposition gp->ffsl

T63L47 (synchronized measurement on blizzard)



From the bottom:



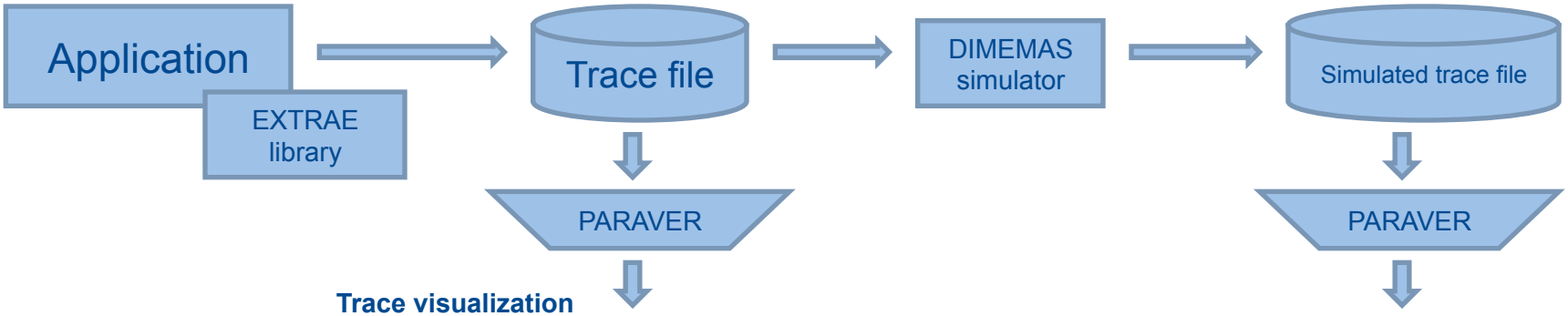
UNDERSTAND WHAT IS GOING ON

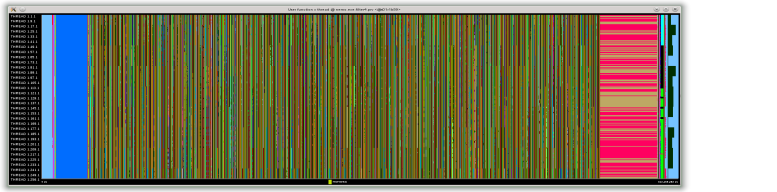
BSC Performance tools

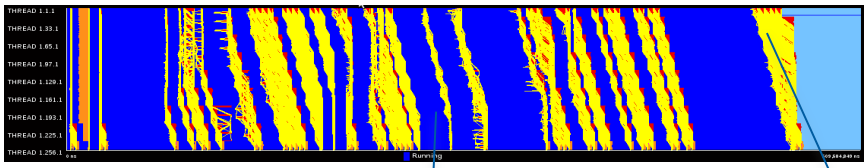
Barcelona Supercomputing Center
Centro Nacional de Supercomputación
EXCELENCIA SEVERO OCHOA

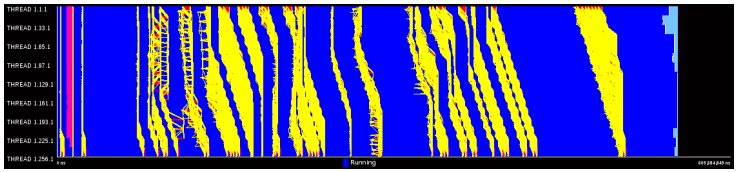
BSC-CNS performance tools:

- EXTRAE: Trace generation package.
- PARAVER: Trace visualization and analysis tool.
- DIMEMAS: Simulation tool for analysis on a configurable target platform.







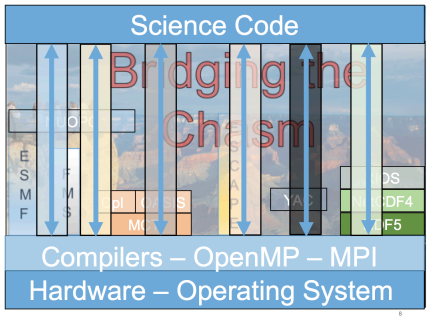


DIMEMAS generated trace. Target = ideal machine

Blue regions = computation

Yellow lines = comunication

IT’S A PEOPLE THING



WHAT MAKES A SUCCESSFUL COMMUNITY TOOL?

Talking is not enough!

We need to work better together!

Can we take just one small step forward together?

What are the processes and procedures?

1. Nothing has to be used by everybody
2. What makes something a community code? OASIS Example
    i. It helps to be the only game in town.
    ii. Not very disruptive of existing solutions. Cost of transition/ implementation.
    iii. User support (tools for collaborative development, responsiveness a cost, who pays?) Training. Documentation. (We all support our own tools, but hide the cost?)
    iv. Integrate developments back from the user community (another cost, what benefit?)
    v. Momentum (in one direction, what about evolution)?
    vi. People don't need to see development path if things are relatively satisfactory?
    vii. Reward structures for new innovation in someone else's package?
    viii. Courage to stop, and pick up and use/integrate others?
    ix. Cf MPIM new activity: YAC (why not reintegrate?)

Process?

1. Open Source
2. Open Development
3. Understanding of community/commitment/ timescale?

Separation of concerns between objectives and implementations? What body exists or could be created that could talk about what to do, as opposed to how?

Ongoing
role for
ENES – the community,
and
ESIWACE – (a) project