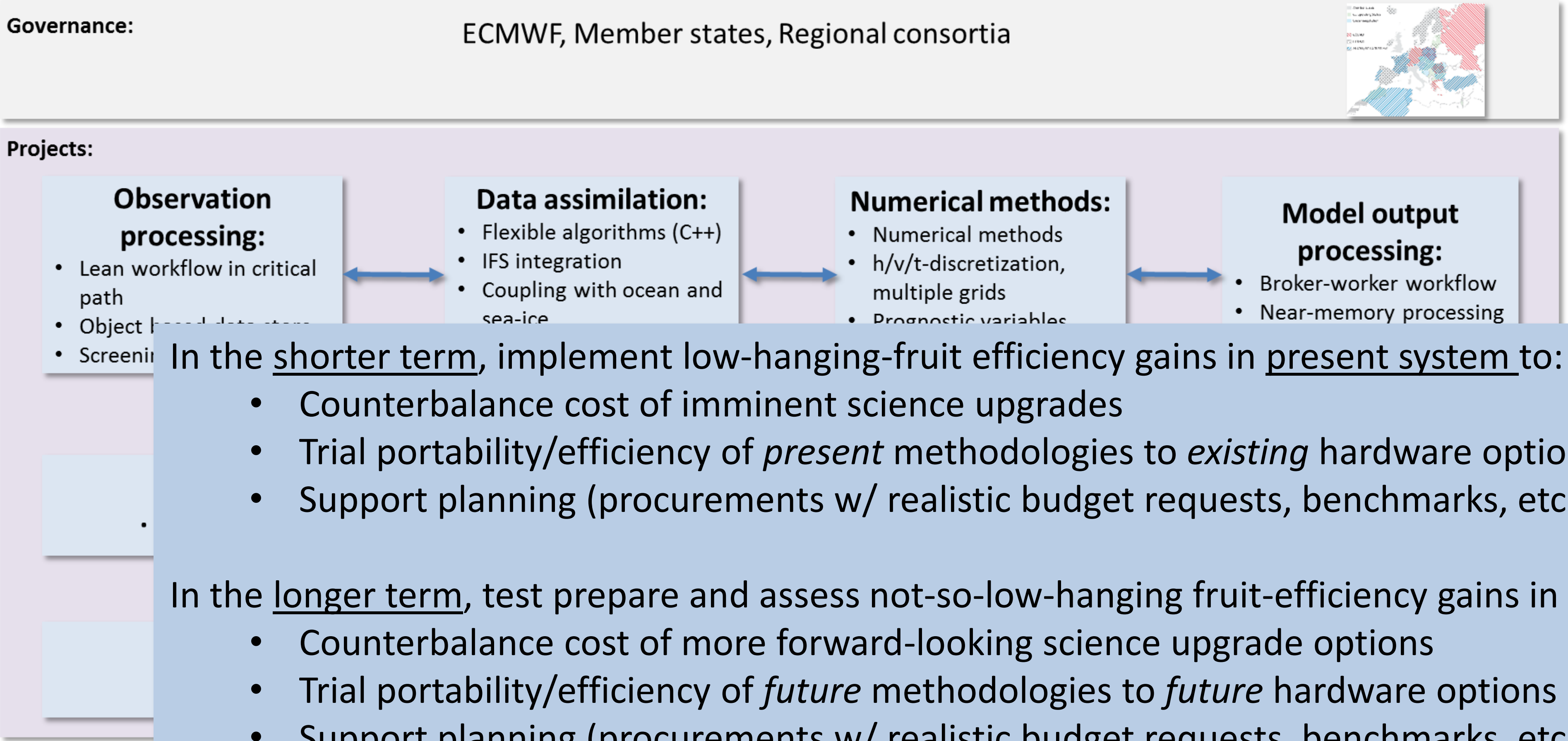
Progress report on ECMWF’s Scalability Programme

Peter Bauer and the Scalability Team (RDxFDxCD, ECMWF)



ECMWF Scalability Programme 1.0

In the <u>shorter term</u>, implement low-hanging-fruit efficiency gains in <u>present system</u> to:

- Counterbalance cost of imminent science upgrades
- Trial portability/efficiency of *present* methodologies to *existing* hardware options
- Support planning (procurements w/ realistic budget requests, benchmarks, etc.)

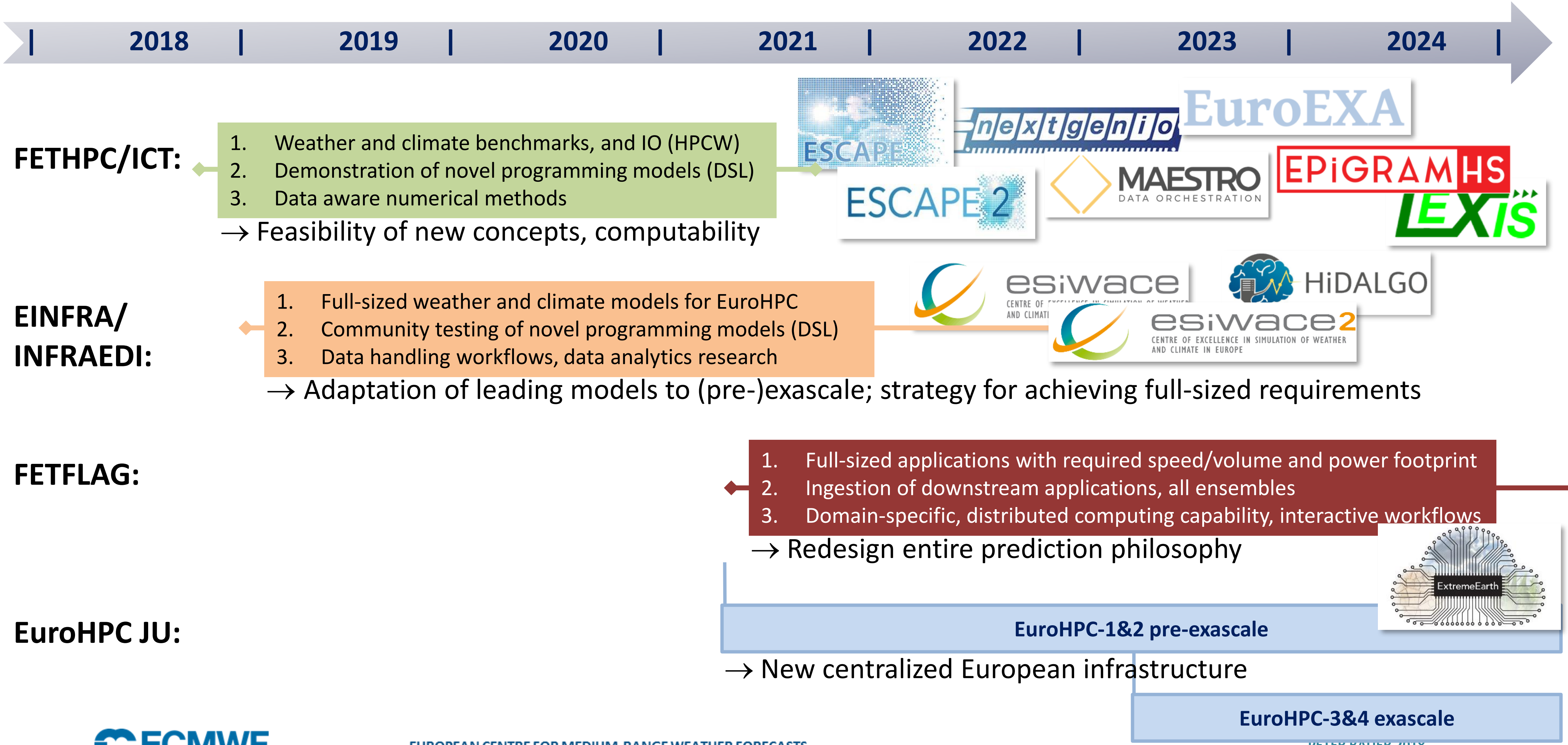
In the <u>longer term</u>, test prepare and assess not-so-low-hanging fruit-efficiency gains in <u>future system</u> to:

- Counterbalance cost of more forward-looking science upgrade options
- Trial portability/efficiency of *future* methodologies to *future* hardware options
- Support planning (procurements w/ realistic budget requests, benchmarks, etc.)

ECMWF

The implications of fulfilling short-term and long-term needs are entirely different!

Weather & climate computing and data roadmap in H2020



FETHPC/ICT:

1. Weather and climate benchmarks, and IO (HPCW)
2. Demonstration of novel programming models (DSL)
3. Data aware numerical methods

→ Feasibility of new concepts, computability

EINFRA/ INFRAEDI:

1. Full-sized weather and climate models for EuroHPC
2. Community testing of novel programming models (DSL)
3. Data handling workflows, data analytics research

→ Adaptation of leading models to (pre-)exascale; strategy for achieving full-sized requirements

FETFLAG:

1. Full-sized applications with required speed/volume and power footprint
2. Ingestion of downstream applications, all ensembles
3. Domain-specific, distributed computing capability, interactive workflows

→ Redesign entire prediction philosophy

EuroHPC JU:

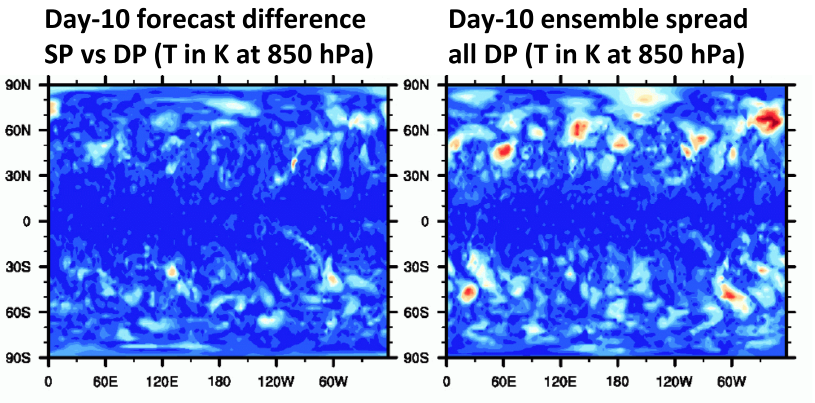
EuroHPC-1&2 pre-exascale

→ New centralized European infrastructure

EuroHPC-3&4 exascale

Low(ish)-hanging fruit: Computing

Funded by the European Union


Day-10 forecast difference SP vs DP (T in K at 850 hPa)

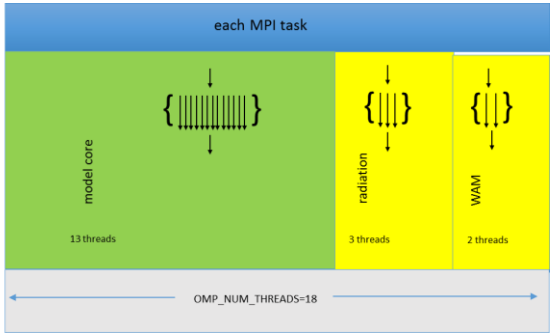
Day-10 ensemble spread all DP (T in K at 850 hPa)

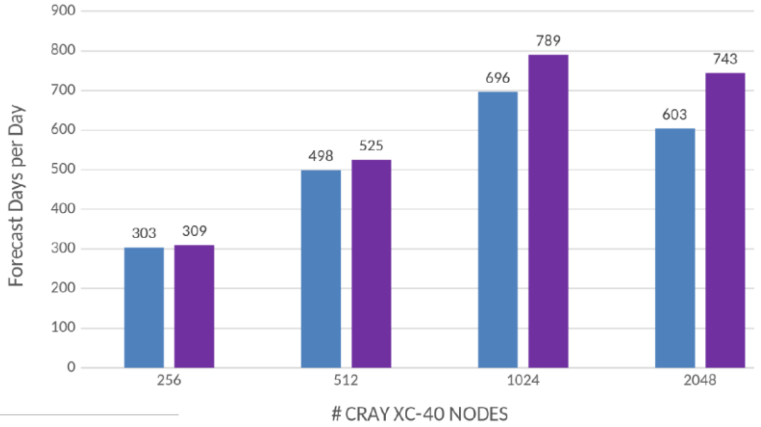
Precision:

- running IFS with single precision arithmetics can save 40% of runtime, IFS-ST offers options like precision by wavenumber, only for LT, in semi-implicit solver;
- storing ensemble model output at reduced precision can save 67% of data volume;

Concurrency:

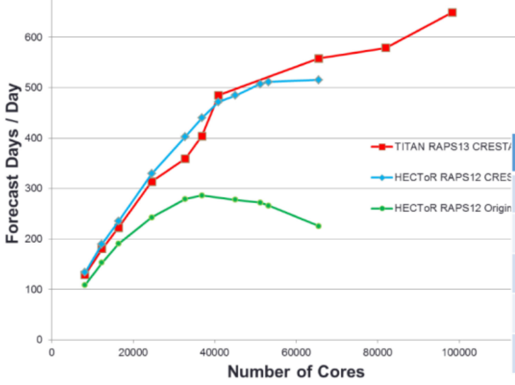
- allocating threads/task (/across tasks) to model components like radiation or waves can save 20% (gain increases with resolution);
- implementation is cumbersome;





Overlapping communication & computing:

- through programming models (Fortran co-array vs GPI2 vs MPI), gave substantial gains on Titan w/Gemini,
- on XC-30/40 w/ Aries there is no overall performance benefit over default MPI implementation;



Tasks x threads	Nodes	Experiments	Forecast days/day
2160Tx12t	360	control	**1104.9**
2160Tx12t	360	control	1116.1
2160Tx12t	360	coarray2	815.6
2160Tx12t	360	coarray2	846.0
2160Tx12t	360	gpi2	788.9

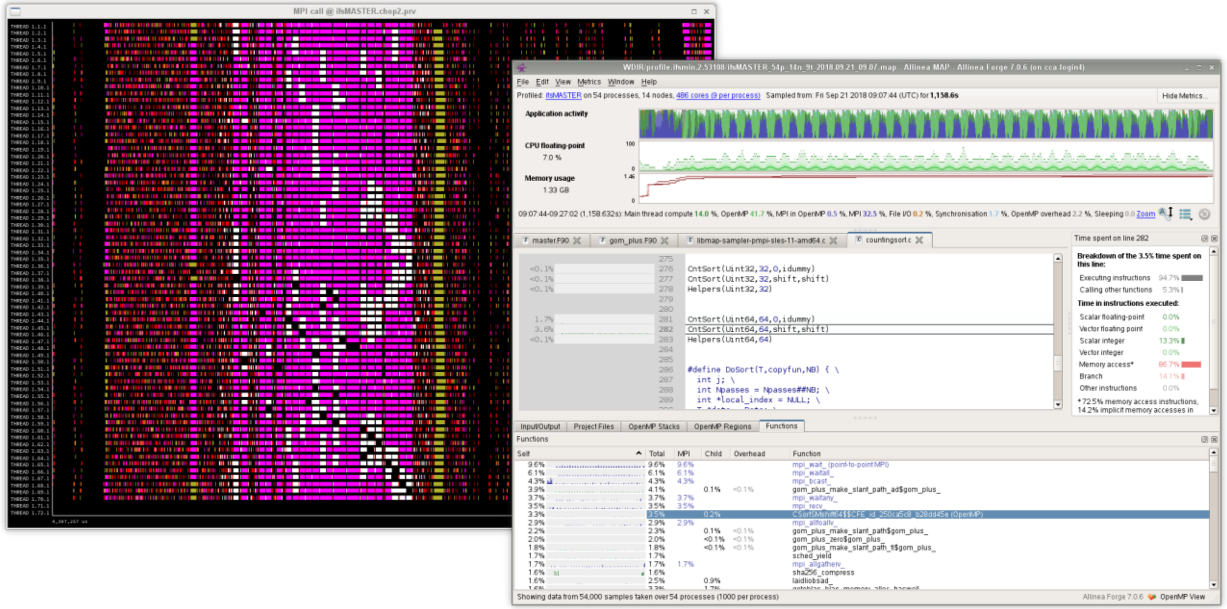
[Düben et al., Vana et al., Mozdzynski et al.]

Low(ish)-hanging fruit: Diagnostics & Architectures

Performance tools:

- Integrate easy-to-use performance tools with IFS, available to all
- ARM Forge MAP, BSC Extrae & Paraver (see POP CoE)

[From Patrick Gillies – check also Mario Acosta's talk on Wednesday!]



WDIR/profile.ifsmin.2.53108/ifsMASTER_54p_14n_9t_2018.09-21_09-07.map - Allinea MAP - Allinea Forge 7.0.6 (on cca-login4)

Profiled: ifsMASTER on 54 processes, 14 nodes, 486 cores (9 per process) Sampled from: Fri Sep 21 2018 09:07:44 (UTC) for 1,158.6s

09:07:44-09:27:02 (1,158.632s): Main thread compute 14.0 %, OpenMP 41.7 %, MPI in OpenMP 0.5 %, MPI 32.5 %, File I/O 0.2 %, Synchronisation 1.7 %, OpenMP overhead 2.2 %, Sleeping 0.0

```
CntSort(Uint32,32,0,idummy)
CntSort(Uint32,32,shift,shift)
Helpers(Uint32,32)


CntSort(Uint64,64,0,idummy)
CntSort(Uint64,64,shift,shift)
Helpers(Uint64,64)


#define DoSort(T,copyfun,NB) { \
  int j; \
  int Npasses = Npasses##NB; \
  int *local_index = NULL; \
```

Time spent on line 282

Breakdown of the 3.5% time spent on this line:

Executing instructions	94.7%
Calling other functions	5.3%

Time in instructions executed:

Scalar floating-point	0.0%
Vector floating point	0.0%
Scalar integer	13.3%
Vector integer	0.0%
Memory access*	86.7%
Branch	14.1%
Other instructions	0.0%

*72.5% memory access instructions, 14.2% implicit memory accesses in

Functions

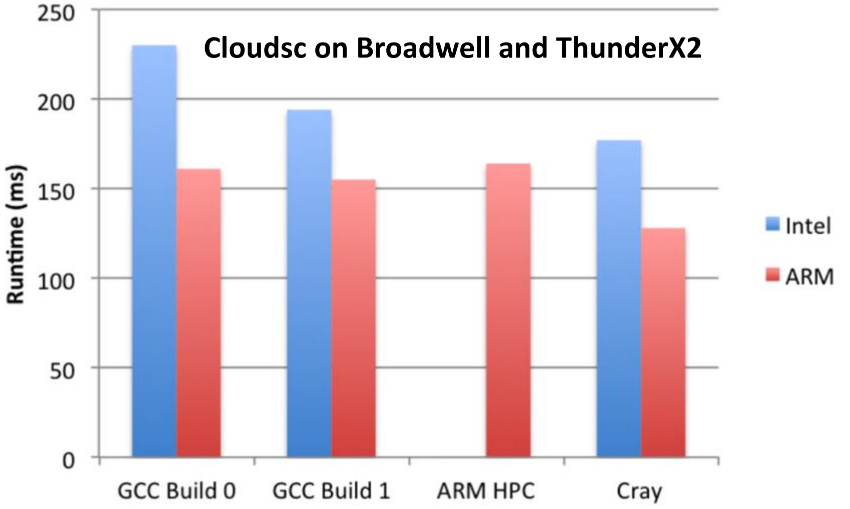
Self	Total	MPI	Child	Overhead	Function
9.6%	9.6%	9.6%			mpi_wait_ (point-to-point MPI)
6.1%	6.1%	6.1%			mpi_waitall_
4.3%	4.3%	4.3%			mpi_bcast_
3.9%	4.1%		0.1%	<0.1%	gom_plus_make_slant_path_ad$gom_plus_
3.7%	3.7%	3.7%			mpi_waitany_
3.5%	3.5%	3.5%			mpi_recv_
3.3%	3.5%		0.2%		CSortSMshift64$$CFE_id_250ca5c8_b28dd45e (OpenMP)
2.9%	2.9%	2.9%			mpi_alltoallv_
2.2%	2.3%		0.1%	<0.1%	gom_plus_make_slant_path$gom_plus_
2.0%	2.0%		<0.1%	<0.1%	gom_plus_zero$gom_plus_
1.8%	1.8%		<0.1%	<0.1%	gom_plus_make_slant_path_tl$gom_plus_
1.7%	1.7%				sched_yield
1.7%	1.7%	1.7%			mpi_allgatherv_
1.6%	1.6%				sha256_compress
1.6%	2.5%		0.9%		laidliobsad_

Showing data from 54,000 samples taken over 54 processes (1000 per process)

Allinea Forge 7.0.6

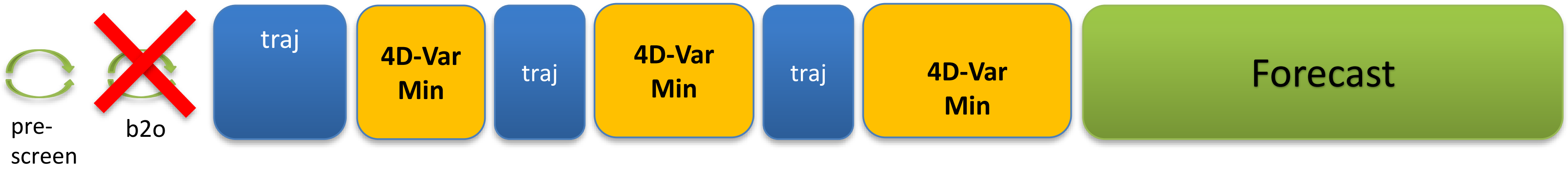
Porting code to other processor types:

- OpenIFS and ESCAPE dwarfs ported to early access nodes - collboration with U Bristol using Isambard Cray platform with Cavium ThunderX2 CPUs
- Long and short-wave MCICA solvers ported to GPU V100 with OpenACC (achieves 85% of peak memory bandwidth on V100) – collaboration with NVIDIA by hackathon for ECMWF staff

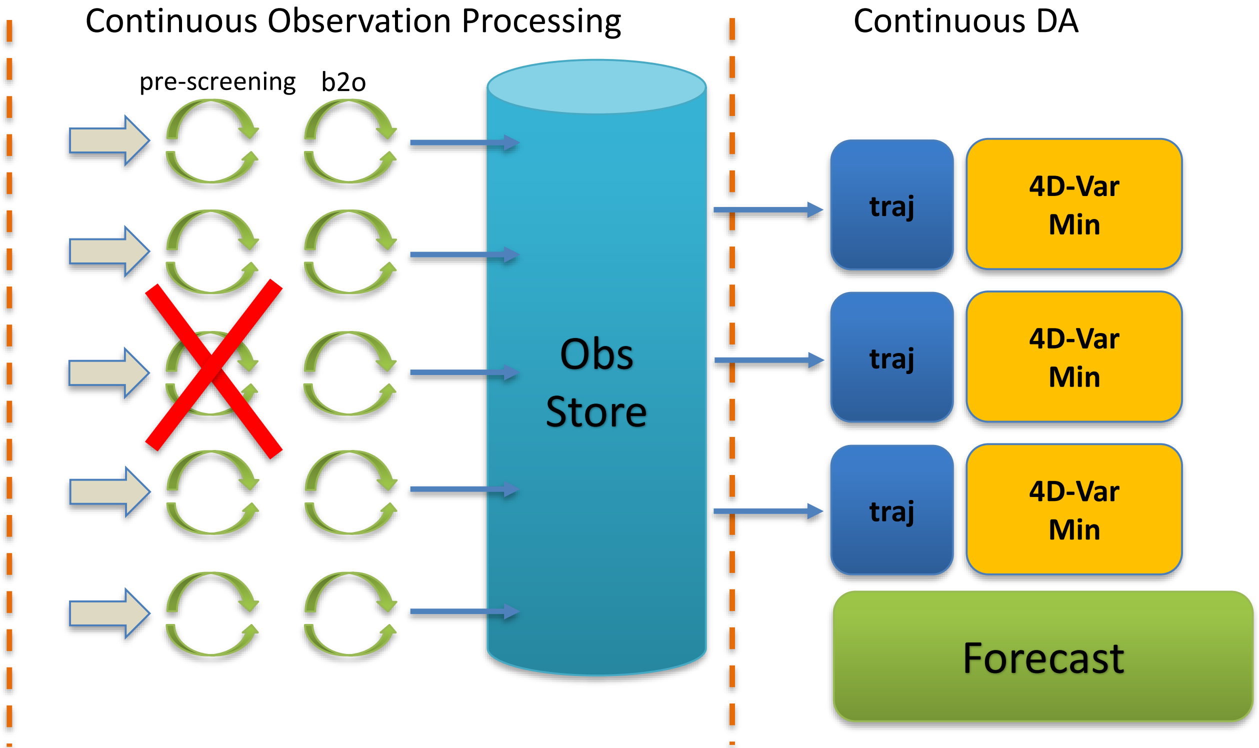


Not-so-low-hanging fruit: Pre-processing

Current processing chain is sequential; a failure at any point leads to delay in forecast production



COPE: Observations pre-screened in small batches as they arrive.
Decoupled system is more robust to failures.



Gains:

- resilience
- 15% cost in critical path

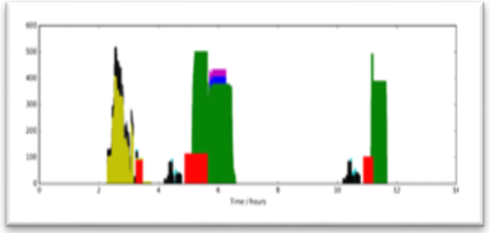
[From Peter Lean – check his talk on Friday!]

Funded by the European Union

nextgenio

Not-so-low-hanging fruit: Benchmarking

- **Kronos tests HPC systems by deploying realistic workloads:**
  1. a workload model is generated from **HPC workload profiling data**
  2. the workload model is then translated (and scaled) into a **schedule of representative and easily-portable applications**
  3. Kronos models and tests **Compute, Interconnect, I/O subsystems**



HPC Workload profiles

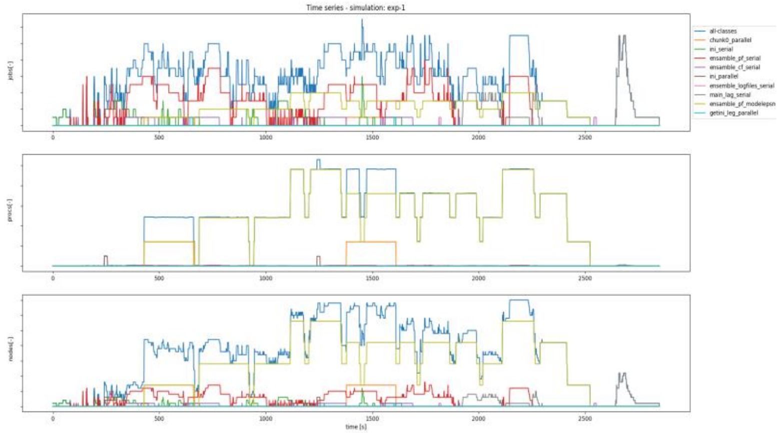
Workload Model

Synthetic Applications

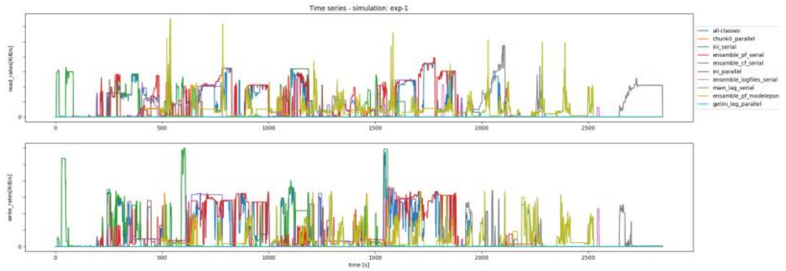
Kronos

HPC prototype to be tested

Post-processing



E.g. Workload execution profiles

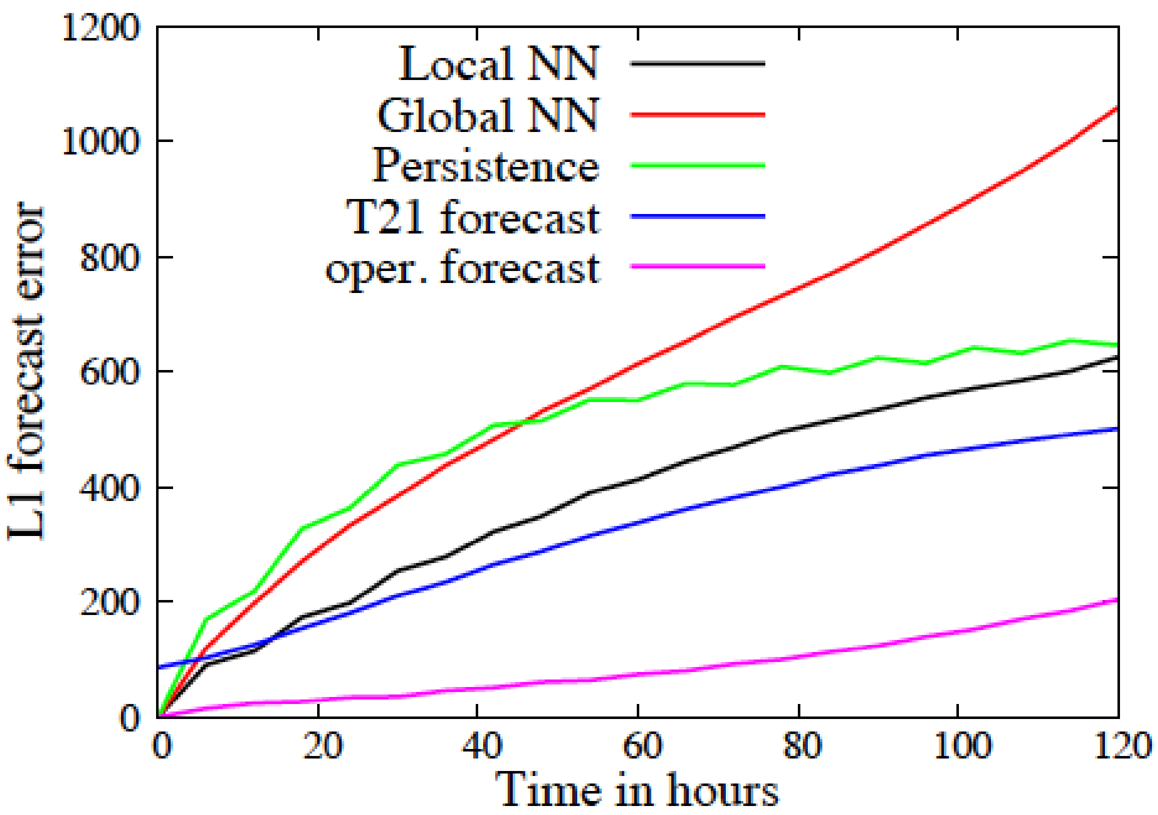


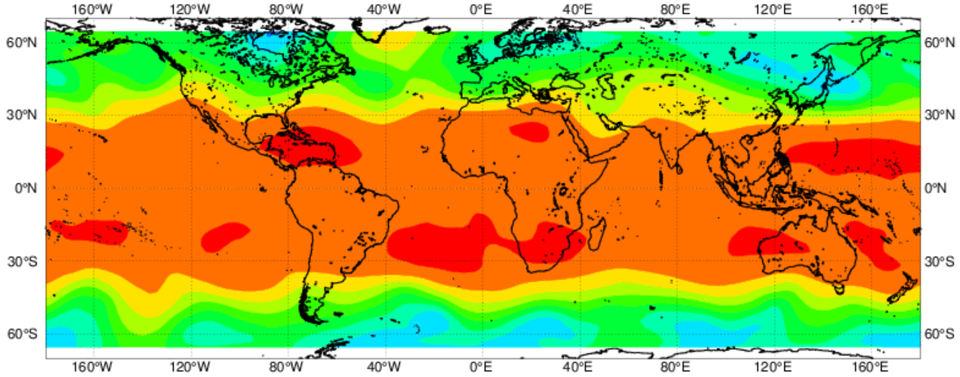
E.g. I/O time-profiles

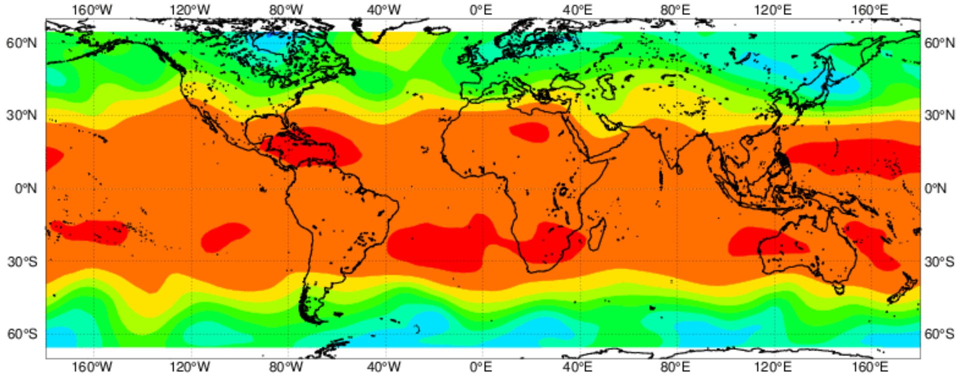
[Antonino Bonanni, Tiago Quintino]

Not-so-low-hanging fruit: AI methods for forecasting

1. Take ERA-5 z500/6º LAT/LON reanalyses/forecasts forecasts = operational forecasts, T21 forecasts, persistence
2. Train NN with truth
3. Run NN forecasts for z500 with all 9x9 grid points predicting tendency = local NN
4. Run NN forecasts for z500 with all grid points predicting tendency = global NN

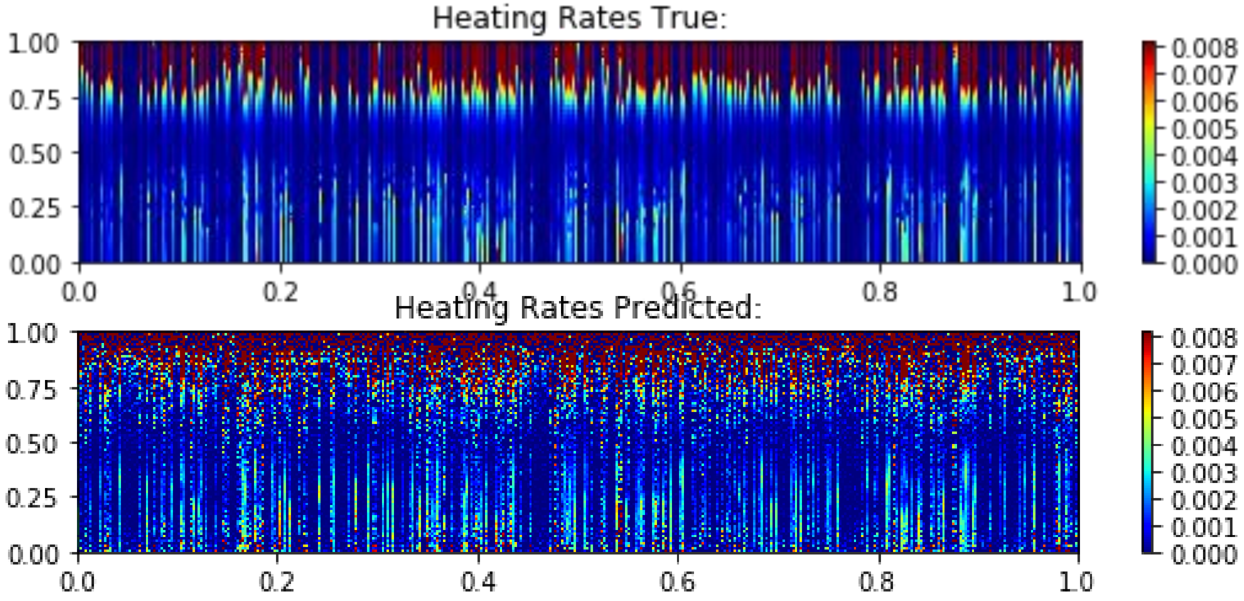






[Düben and Bauer 2018]

Not-so-low-hanging fruit: AI methods for parameterizations



Data Set: 150,000 profiles total (25,000 locations with different solar zenith angles), divided into training=126,000, validation=24,000

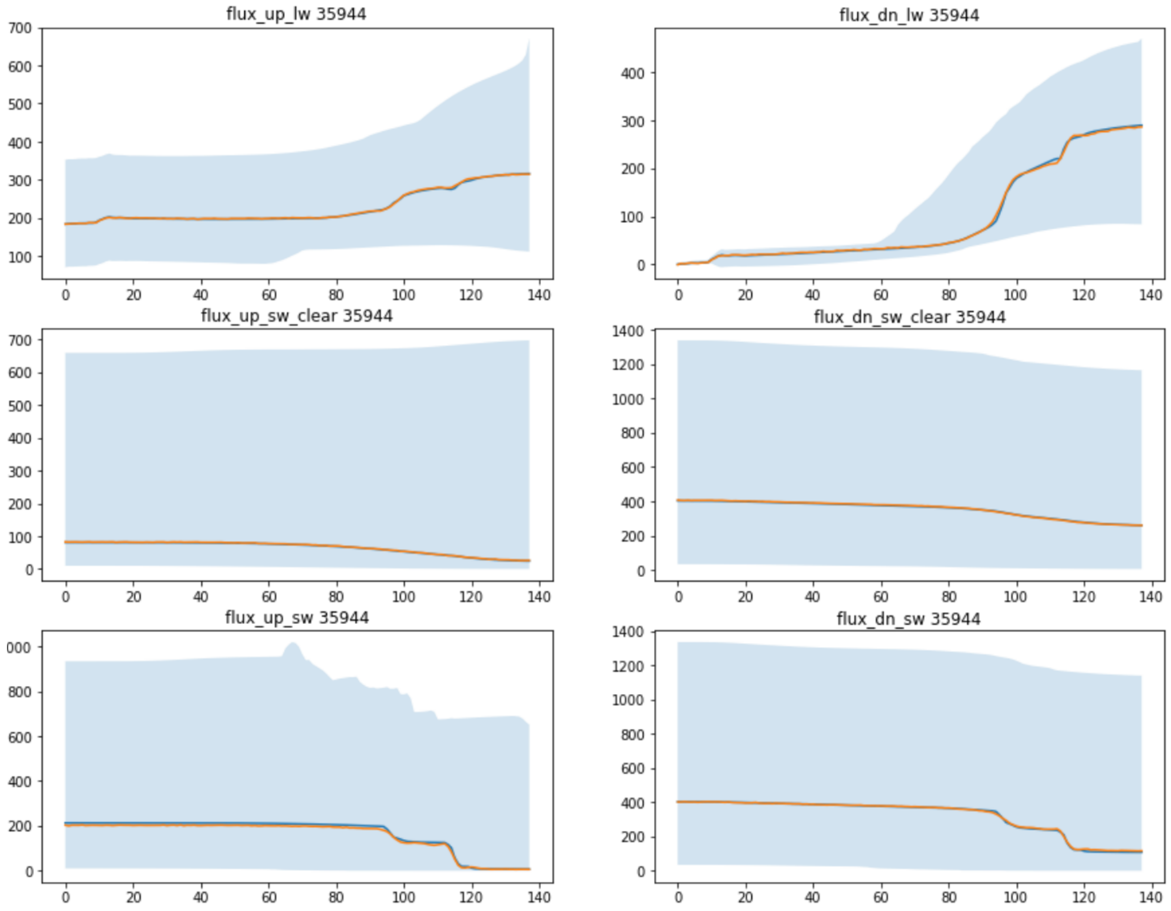
Input to the network: 128 x 137 x 19 (128 batch size, 137 full levels, 19 variables SW clear sky)

Output of the network: 128 x 138 x 2 (up and down flux on each half level)

Network: four 1D convolutional layers followed by two fully connected layers; 194k trainable parameters

Shortwave and longwave flux profiles

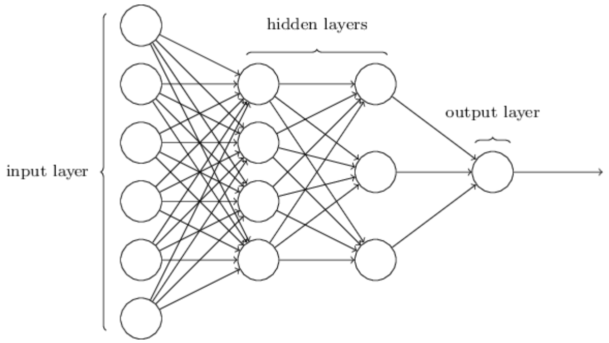
(reference, NN, shading = natural variability)

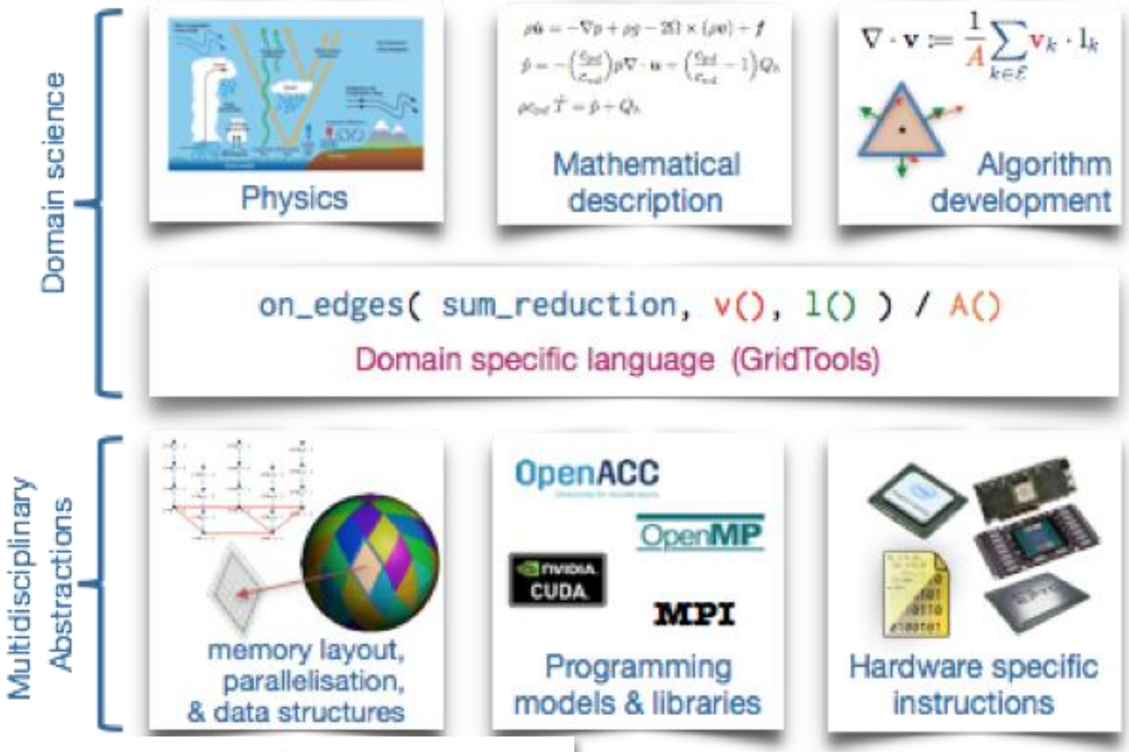


[Christoph Angerer & Jakob Progsch, NVIDIA; Peter Düben, Robin Hogan, Peter Bauer]

Far-hanging fruits: Algorithms – programming - hardware

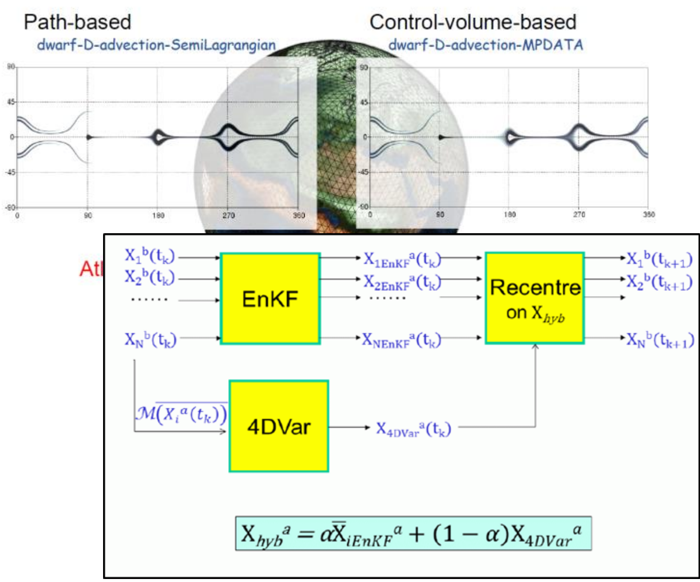
Neural networks



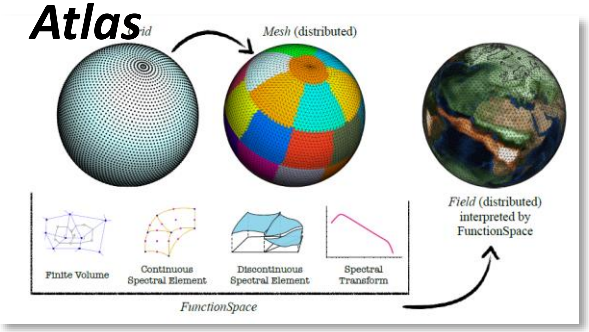

Domain science
Physics
Mathematical description
Algorithm development
$\nabla \cdot \mathbf{v} := \frac{1}{A}\sum_{k\in\mathcal{E}} \mathbf{v}_k \cdot \mathbf{l}_k$
on_edges(sum_reduction, v(), l()) / A()
Domain specific language (GridTools)
Multidisciplinary Abstractions
memory layout, parallelisation, & data structures
OpenACC
OpenMP
NVIDIA CUDA
MPI
Programming models & libraries
Hardware specific instructions

Mathematics&algorithms

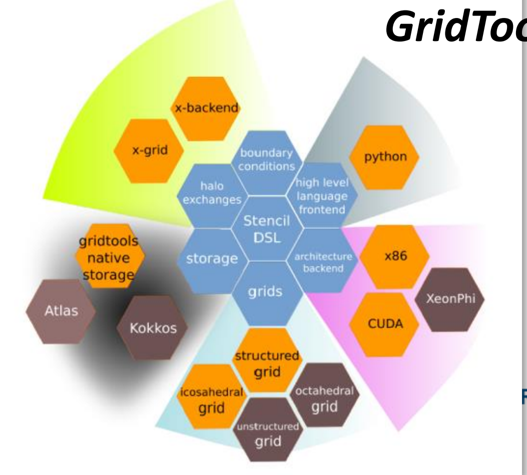
Rossby-Haurwitz test case after 7 days



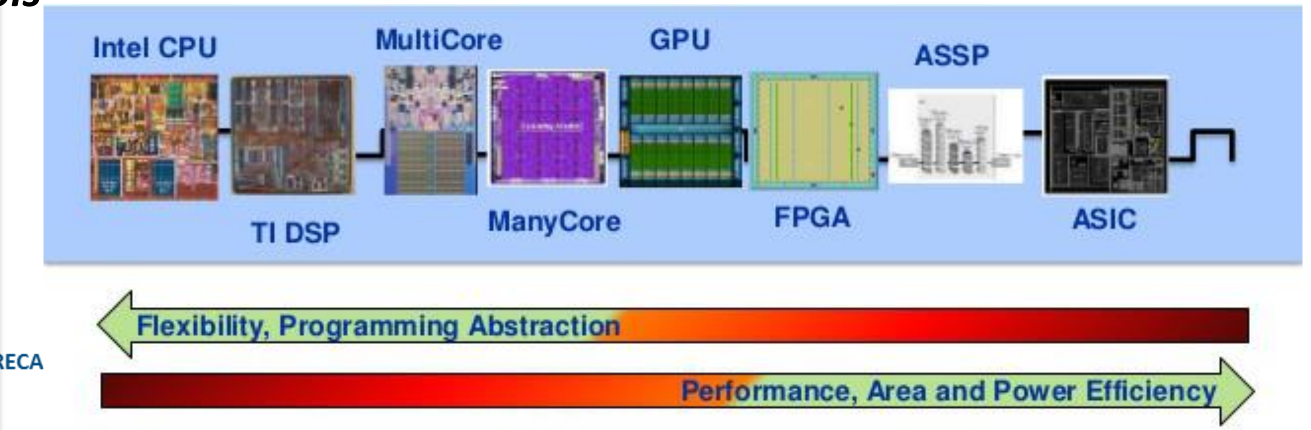
Atlas


Mesh (distributed)
Field (distributed) interpreted by FunctionSpace
Finite Volume
Continuous Spectral Element
Discontinuous Spectral Element
Spectral Transform
FunctionSpace

GridTools



Processors

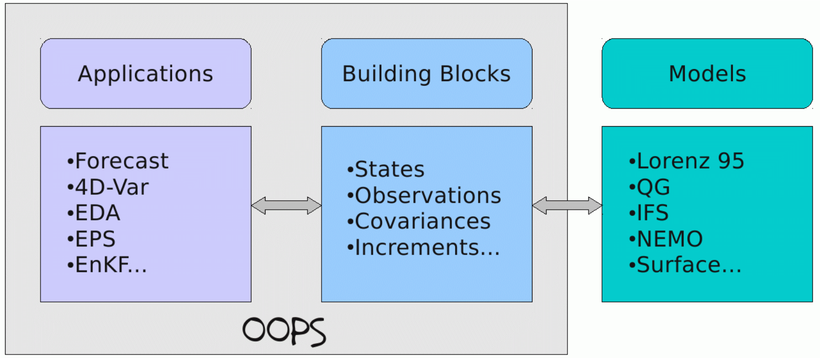

Intel CPU
TI DSP
MultiCore
ManyCore
GPU
FPGA
ASSP
ASIC
Flexibility, Programming Abstraction
Performance, Area and Power Efficiency

Far-hanging fruits: Algorithms – programming - hardware

Algorithmic flexibility equally applicable to:

Data Assimilation

Forecasting



Applications

- •Forecast
- •4D-Var
- •EDA
- •EPS
- •EnKF...

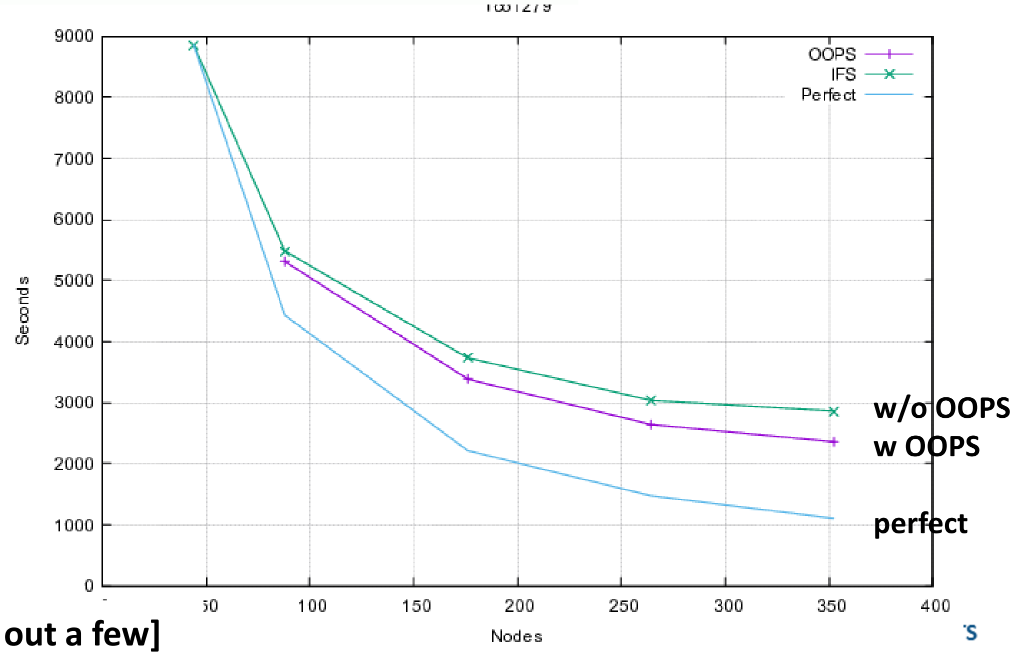
Building Blocks

- •States
- •Observations
- •Covariances
- •Increments...

Models

- •Lorenz 95
- •QG
- •IFS
- •NEMO
- •Surface...

OOPS

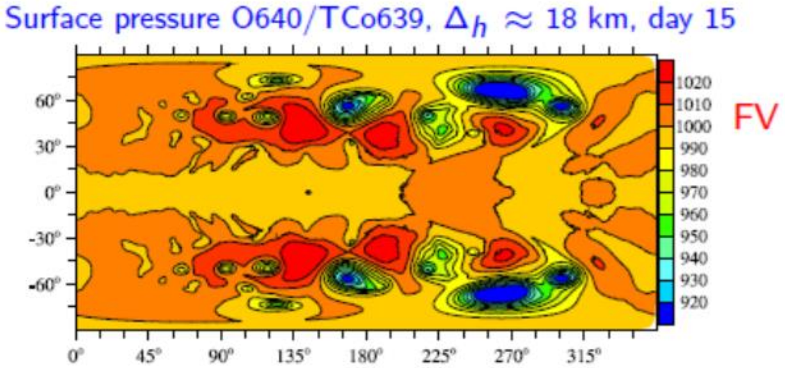


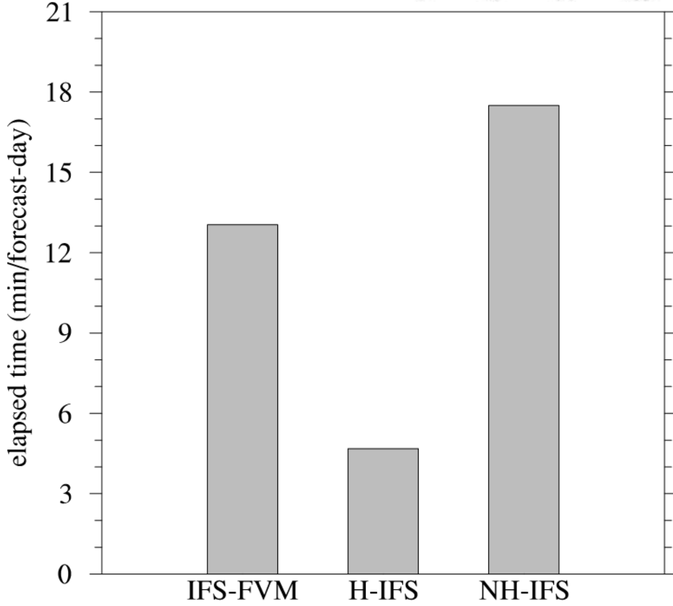
w/o OOPS

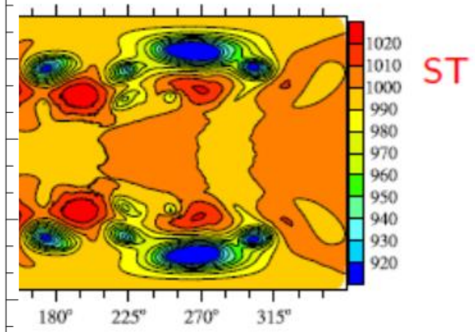
w OOPS

perfect

[Too many to single out a few]





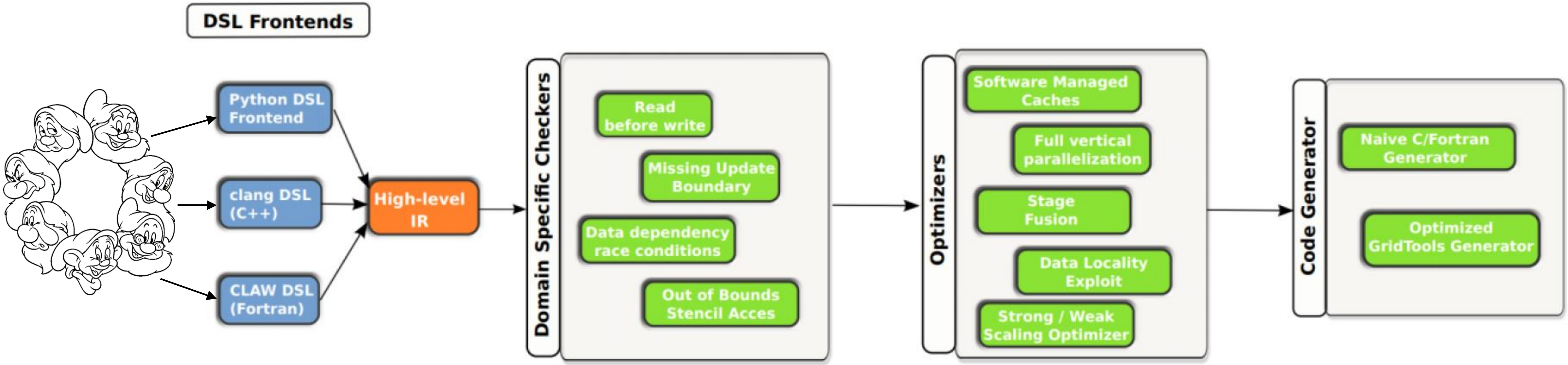


O1280/TCo1279 and L137 using dry dycore on 350 nodes of ECMWF's Cray XC40

[Christoph Kühnlein, Piotr Smolarkiewicz

Far-hanging fruits: Algorithms – programming - hardware

The ESCAPE 2 DSL tool-chain …



DSL Frontends

Python DSL Frontend

clang DSL (C++)

CLAW DSL (Fortran)

High-level IR

Domain Specific Checkers

Read before write

Missing Update Boundary

Data dependency race conditions

Out of Bounds Stencil Acces

Optimizers

Software Managed Caches

Full vertical parallelization

Stage Fusion

Data Locality Exploit

Strong / Weak Scaling Optimizer

Code Generator

Naive C/Fortran Generator

Optimized GridTools Generator

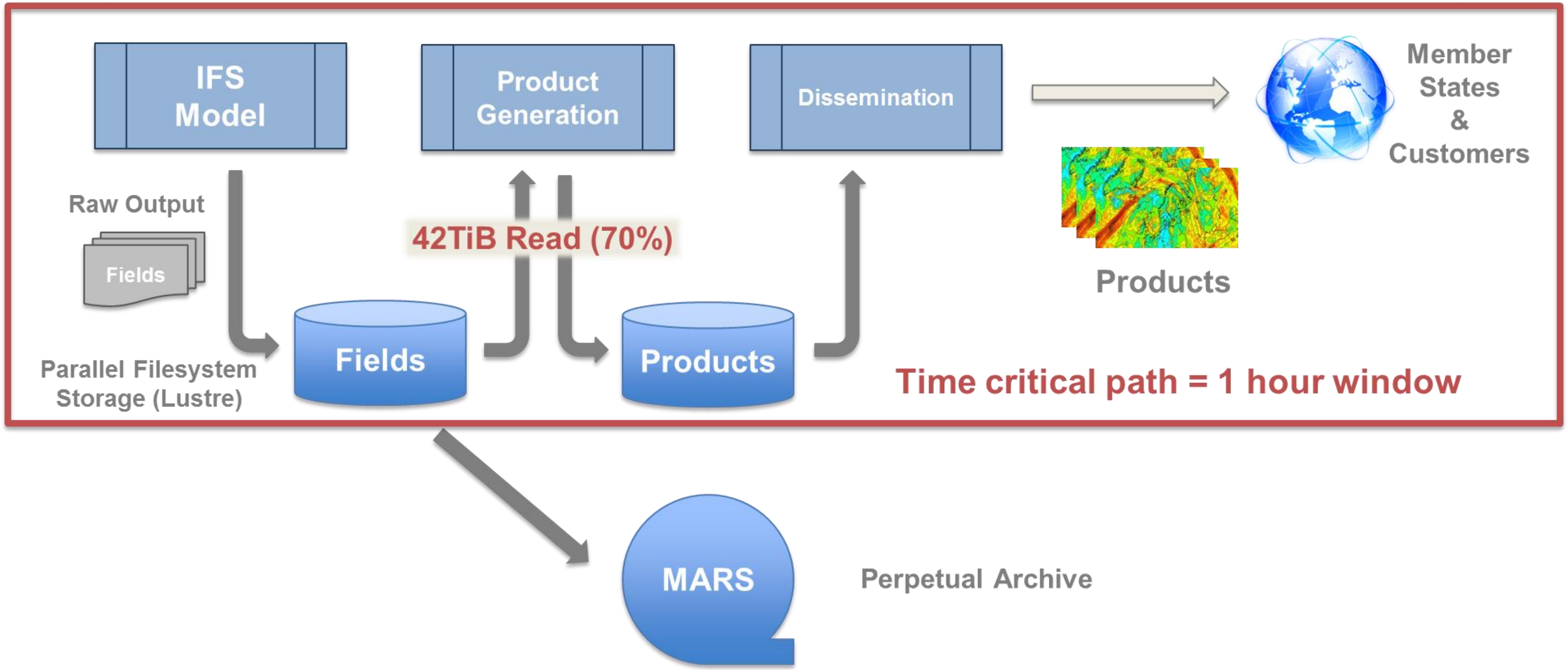
… supported by the EURO EXA toolchain:

1. Generate single-column abstraction code for physics dwarfs using Loki (=Python code transformation)
2. Generate GPU-code using CLAW
3. Generate prototype C-kernel for initial FPGA porting

[Michael Lange, Olivier Marsden]

Funded by the European Union

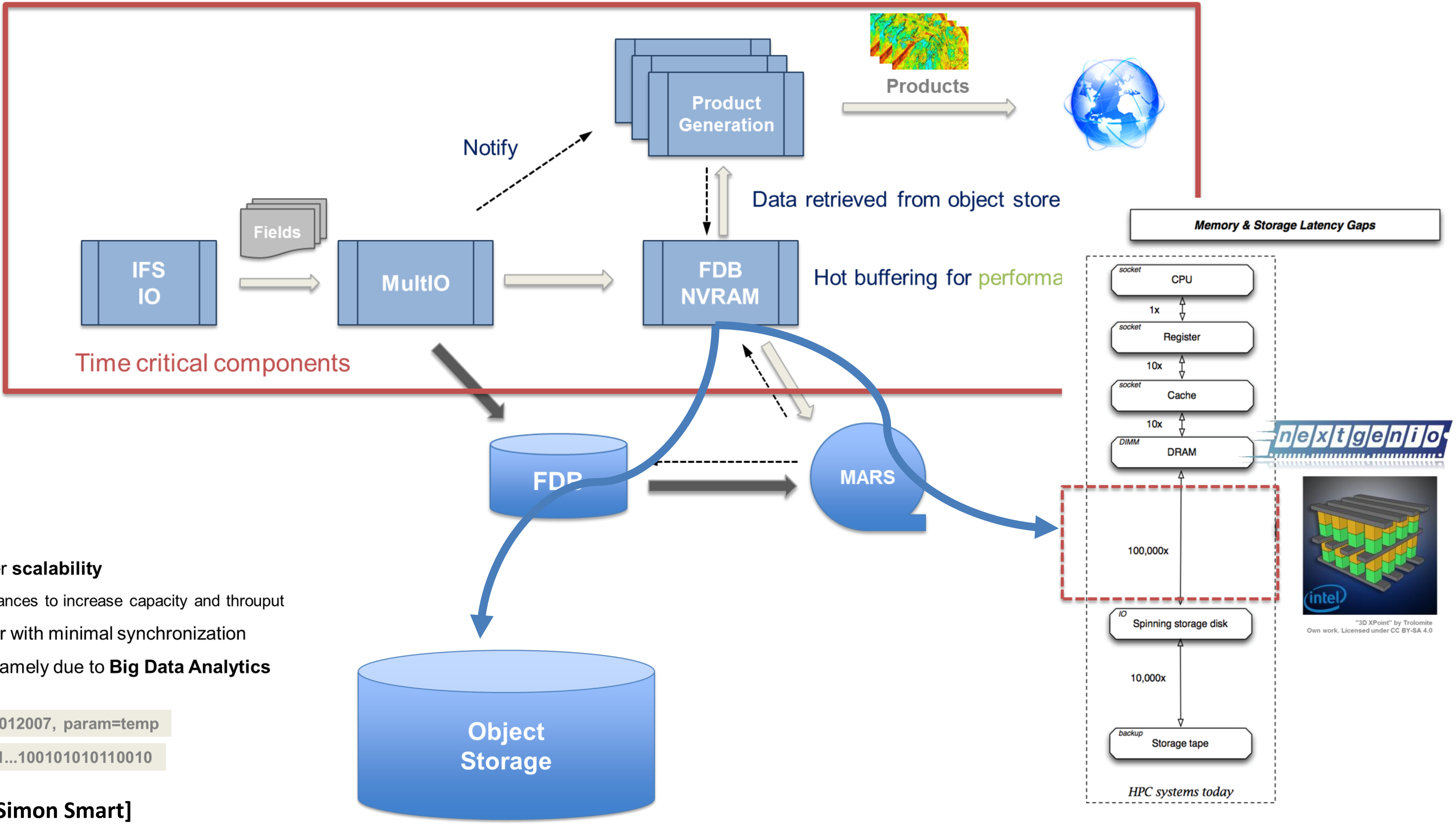
Examples of far-hanging fruit: Post-processing



	Model	Model + I/O	Model + I/O + PGen
Nodes	2440	2776	2926
Run time [s]	5765	6749	7260
Relative	-	+ 17%	+ 26%

[Tiago Quintino, Simon Smart]

Examples of far-hanging fruit: Post-processing



"3D XPoint" by Trolomite
Own work. Licensed under CC BY-SA 4.0

- Key-Value stores offer **scalability**
  - Just add more instances to increase capacity and throuput
- **Transaction** behavior with minimal synchronization
- Growing popularity, namely due to **Big Data Analytics**

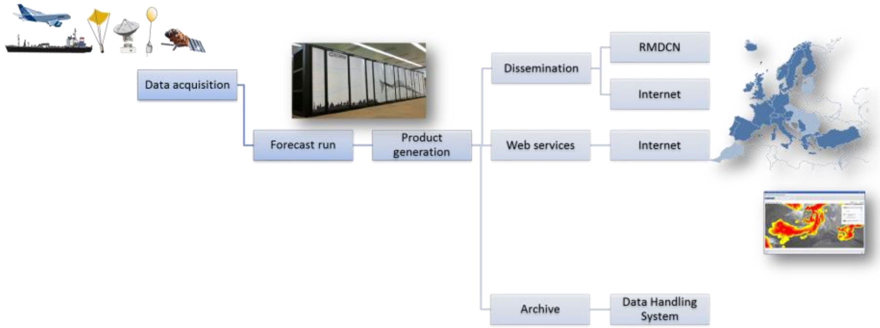
Key: date=12012007, param=temp

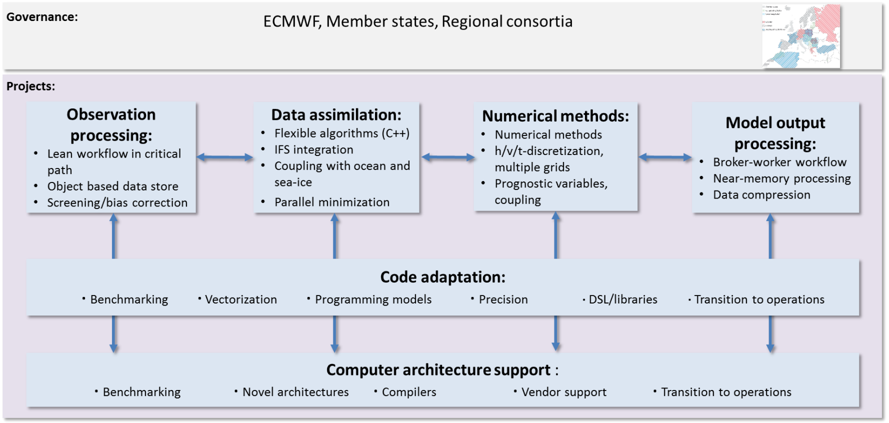
Value: 101001...100101010110010

[Tiago Quintino, Simon Smart]

ECMWF Scalability Programme 2.0

Scalability Programme 1.0





Governance: ECMWF, Member states, Regional consortia

Projects:

Observation processing:
- Lean workflow in critical path
- Object based data store
- Screening/bias correction

Data assimilation:
- Flexible algorithms (C++)
- IFS integration
- Coupling with ocean and sea-ice
- Parallel minimization

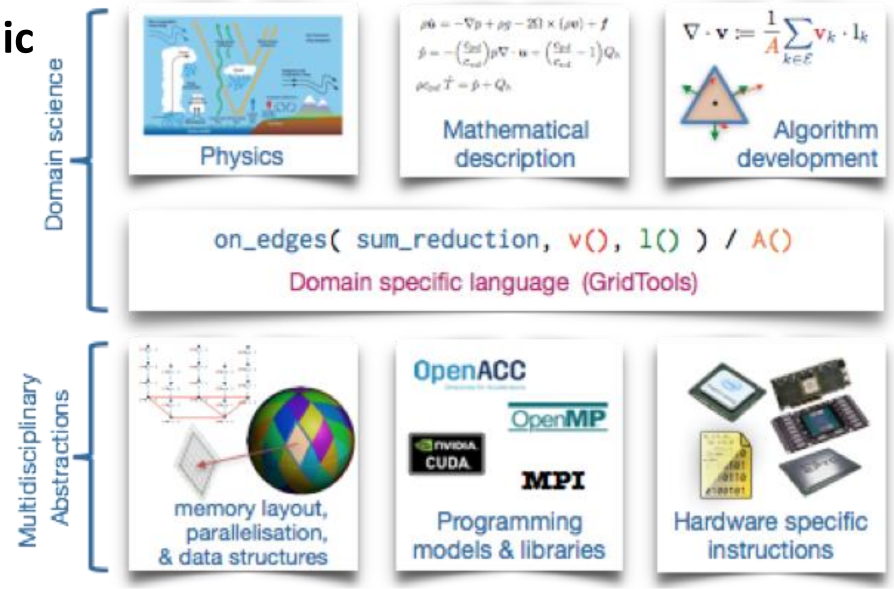
Numerical methods:
- Numerical methods
- h/v/t-discretization, multiple grids
- Prognostic variables, coupling

Model output processing:
- Broker-worker workflow
- Near-memory processing
- Data compression

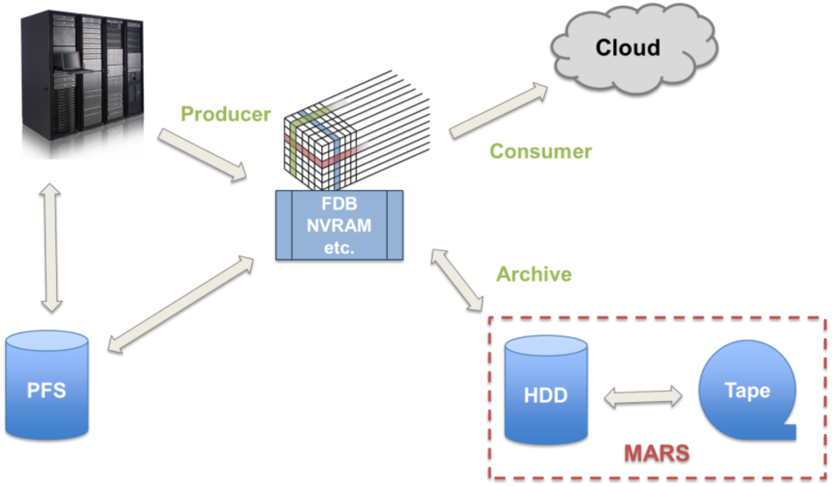
Code adaptation:
- Benchmarking
- Vectorization
- Programming models
- Precision
- DSL/libraries
- Transition to operations

Computer architecture support :
- Benchmarking
- Novel architectures
- Compilers
- Vendor support
- Transition to operations

Co-development of algorithmic options, programming and hardware:


Domain science
Physics
Mathematical description
Algorithm development
on_edges(sum_reduction, v(), l()) / A()
Domain specific language (GridTools)
Multidisciplinary Abstractions
memory layout, parallelisation, & data structures
OpenACC
OpenMP
NVIDIA CUDA
MPI
Programming models & libraries
Hardware specific instructions

Co-development of flexible workflows, object data stores and hardware (cloud aware):



There is an opportunity to take this to the extreme!

FET FLAGSHIPS

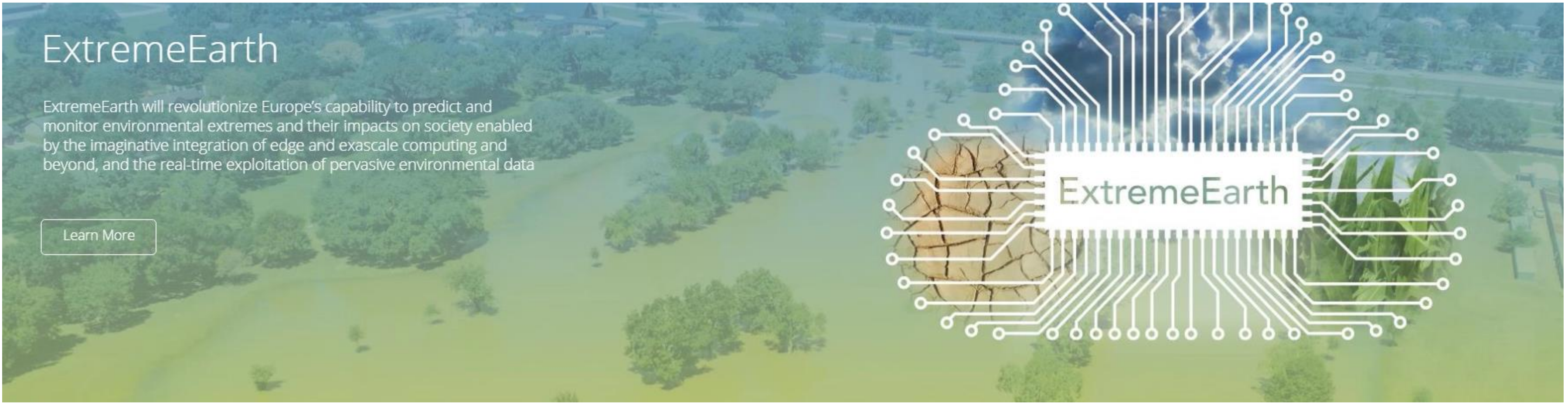
Human Brain Project

GRAPHENE FLAGSHIP

Future and Emerging Technology Flagships are:

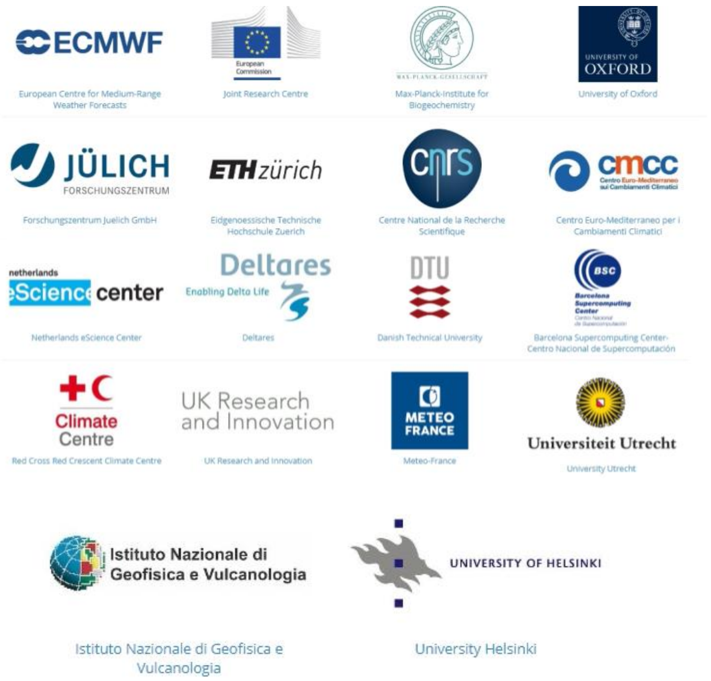
"… science- and technology-driven, large-scale, multidisciplinary research initiatives built around a visionary unifying goal ... tackle grand science and technology challenges … strong and broad basis for future innovation and economic exploitation … novel benefits for society of a potential high impact … long-term and sustained effort."

The *ExtremeEarth* proposal: **www.extremeearth.eu**



ExtremeEarth

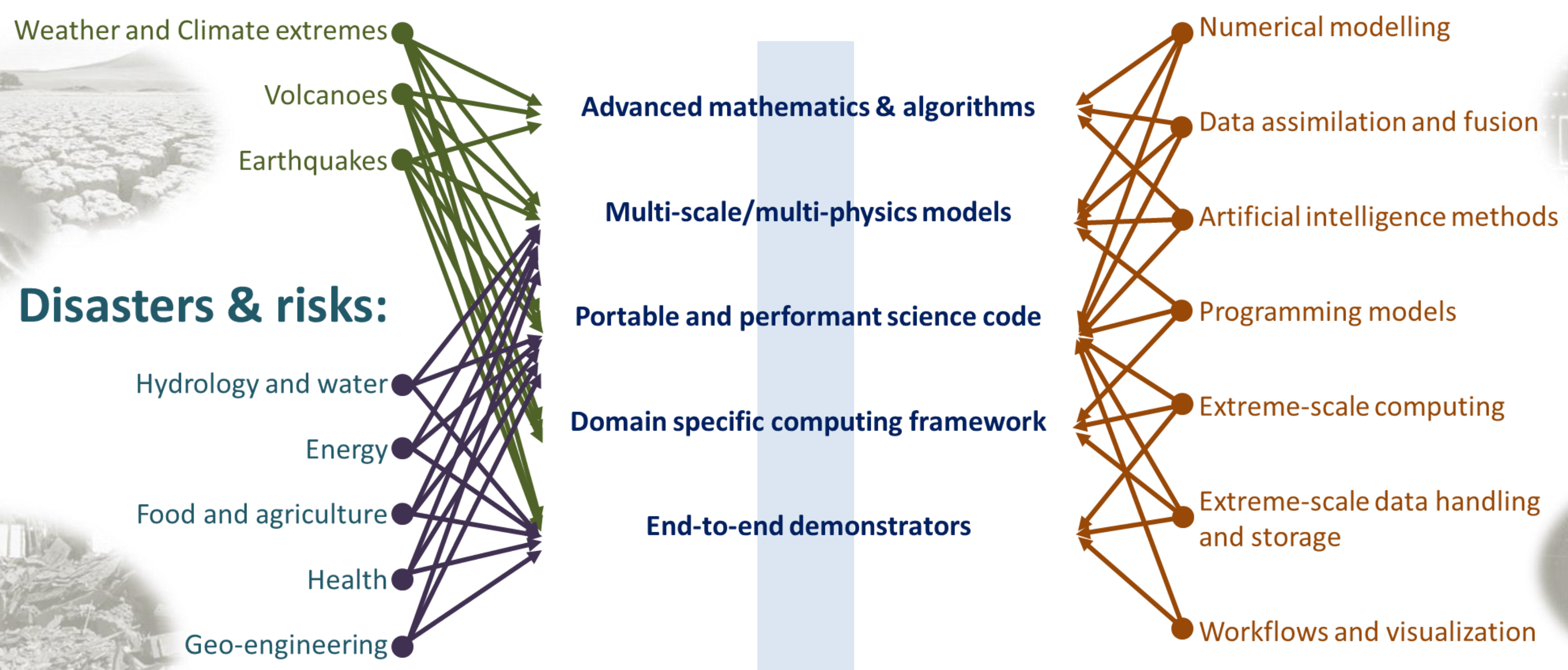
ExtremeEarth will revolutionize Europe's capability to predict and monitor environmental extremes and their impacts on society enabled by the imaginative integration of edge and exascale computing and beyond, and the real-time exploitation of pervasive environmental data



European Centre for Medium-Range Weather Forecasts	Joint Research Centre	Max-Planck-Institute for Biogeochemistry	University of Oxford
Forschungszentrum Juelich GmbH	Eidgenoessische Technische Hochschule Zuerich	Centre National de la Recherche Scientifique	Centro Euro-Mediterraneo per i Cambiamenti Climatici
Netherlands eScience Center	Deltares	Danish Technical University	Barcelona Supercomputing Center-Centro Nacional de Supercomputación
Red Cross Red Crescent Climate Centre	UK Research and Innovation	Meteo-France	University Utrecht
Istituto Nazionale di Geofisica e Vulcanologia	University Helsinki		

ExtremeEarth

Co-Design



Science:

- Weather and Climate extremes
- Volcanoes
- Earthquakes

Disasters & risks:

- Hydrology and water
- Energy
- Food and agriculture
- Health
- Geo-engineering

Advanced mathematics & algorithms

Multi-scale/multi-physics models

Portable and performant science code

Domain specific computing framework

End-to-end demonstrators

Technology:

- Numerical modelling
- Data assimilation and fusion
- Artificial intelligence methods
- Programming models
- Extreme-scale computing
- Extreme-scale data handling and storage
- Workflows and visualization

Ultra high-resolution Earth-system physics-impact <u>modelling capability</u>

Extreme-scale Earth-system <u>data management capability</u>

Extreme-scale Earth-system <u>computing capability</u>

Interactive Earth-system <u>information system capability</u>

Future forecasting

Incremental:

```
do while (skill .ne. good_enough)
       model%resolution       = model%resolution      / model%dresolution
       model%complexity       = model%complexity      * model%dcomplexity
       ensemble%size          = ensemble%size         * ensemble%dsize
       downstream%application = downstream%application + 1

       call performance (model, ensemble, downstream, speed)
       call translate   (model, ensemble, downstream, speed, software, hardware)

       do while (speed .ne. fast_enough)
              call add_funding      (bucks, software, hardware)
              call add_optimization (software, hardware, speed)
              call add_processors   (software, hardware, speed)

              if (bucks .gt. budget) abort
       end do
       call science (model, ensemble, downstream, skill)
end do
```

Radical:

```
call extreme_earth
```