Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Federal Department of Home Affairs FDHA
Federal Office of Meteorology and Climatology MeteoSwiss

Designs for Efficient Weather & Climate Models

Carlos Osuna[1], Oliver Fuhrer[1], Xavier Lapillonne[1], Mauro Bianco[2], Paolo Crosetto[3], Thomas Schulthess[2]

[1]Federal Institute of Meteorology and Climatology MeteoSwiss
[2]Swiss National Supercompuing Centre CSCS, Lugano
[3]Centre for Climate Systems Modeling C2SM, ETH Zurich

.

Current operational system @ Meteoswiss

ECMWF-Model

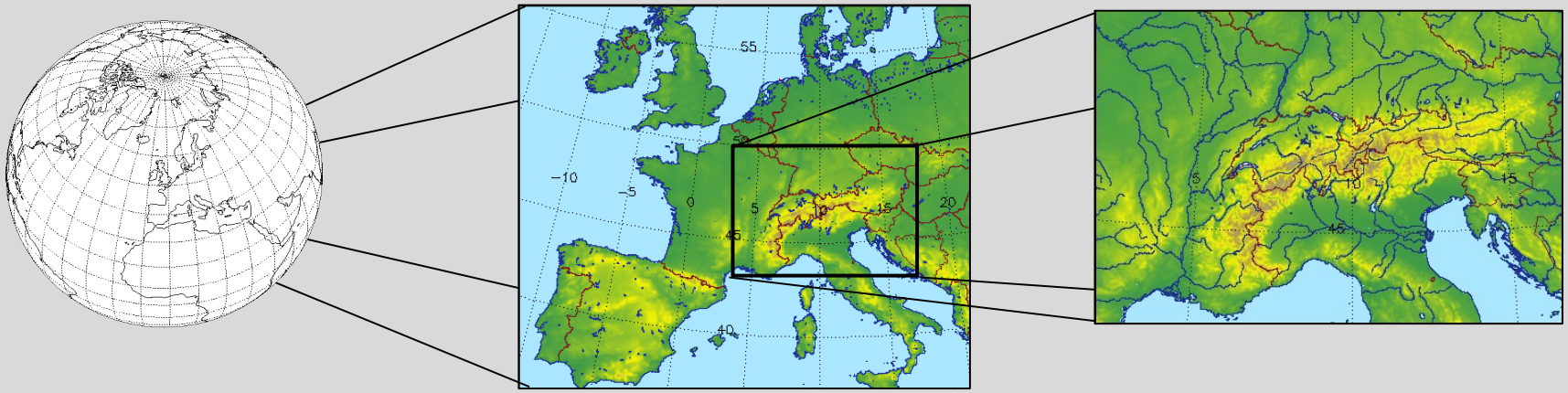
16 km gridspacing
2 x per day 10 day forecast

COSMO-7

Δx = 6.6 km, Δt = 60 s
393 x 338 x 60 cells
3 x per day 72 h forecast

COSMO-2

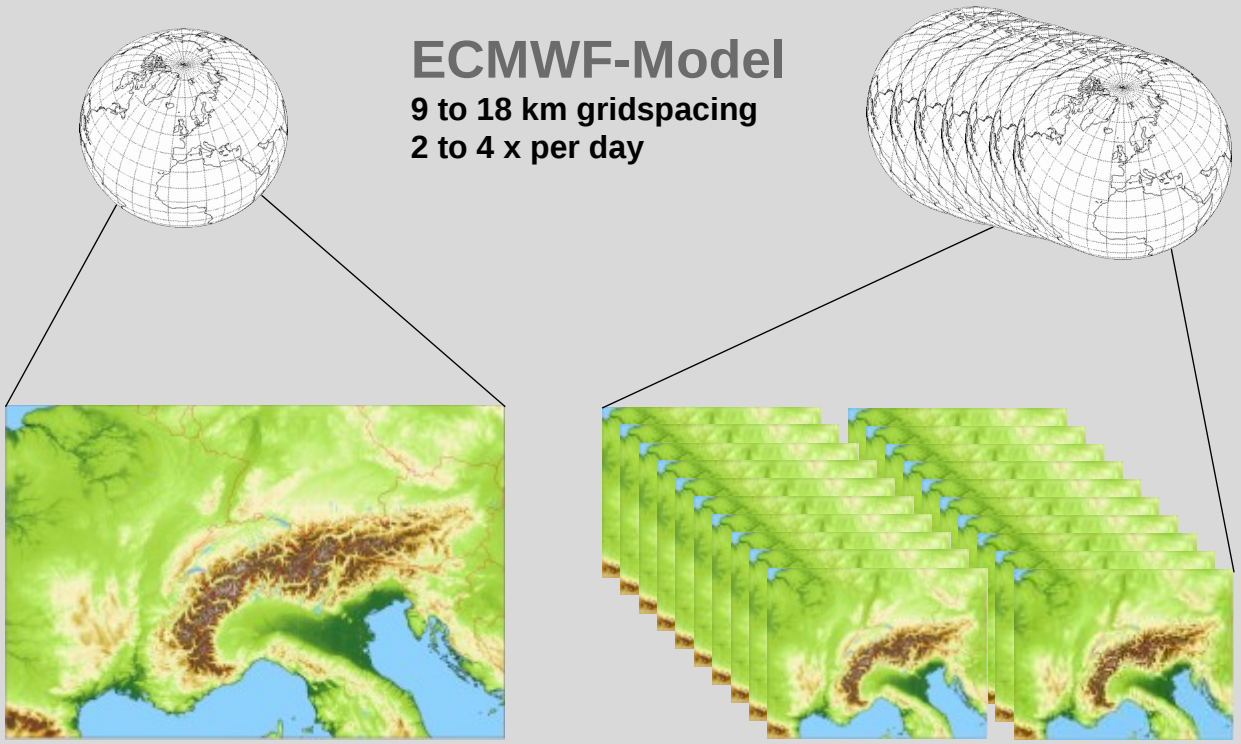
Δx = 2.2 km, Δt = 20 s
520 x 350 x 60 cells
7 x per day 33 h forecast
1 x per day 45 h forecast



Next-generation system

ECMWF-Model

9 to 18 km gridspacing
2 to 4 x per day



COSMO-1

Δx = 1.1 km, Δt = 10 s
1158 x 774 x 80 cells
8 x per day
1 - 2 d forecast

COSMO-E

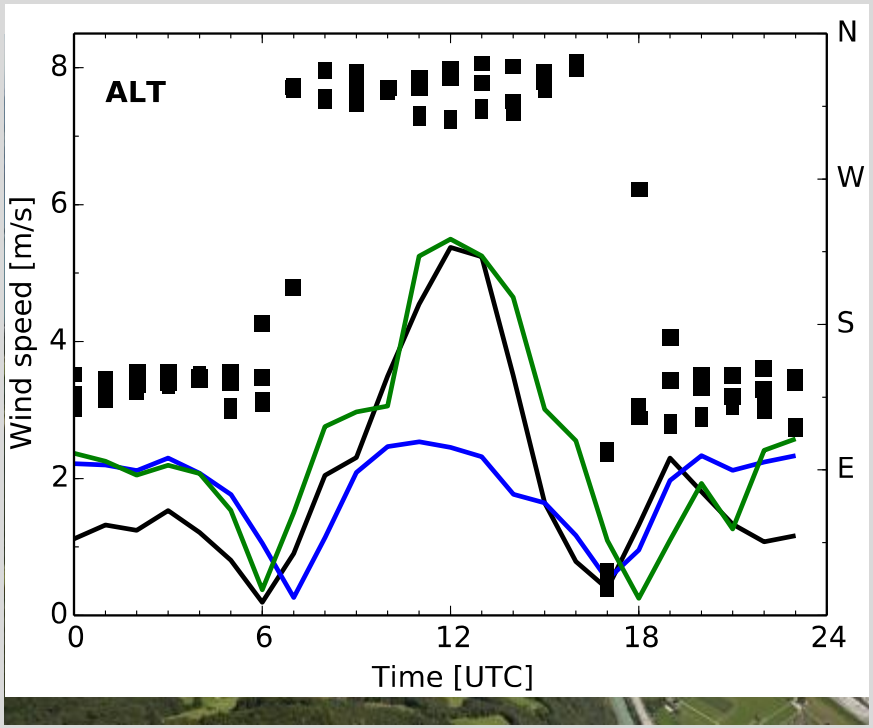
Δx = 2.2 km, Δt = 20 s
582 x 390 x 60 cells
2 x per day
5 d forecast
21 members

Ensemble data assimilation: LETKF

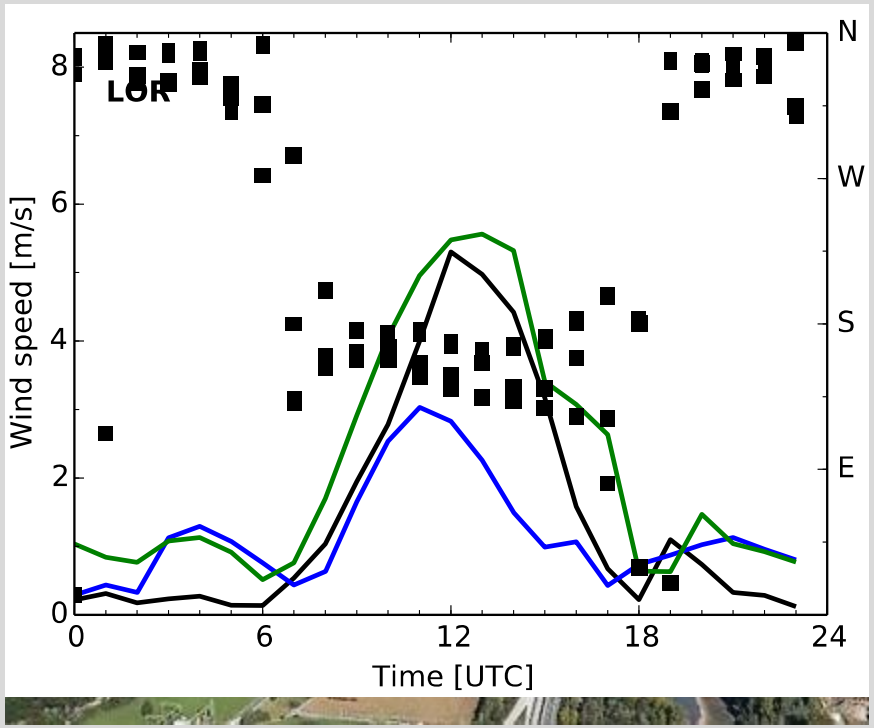
Benefit of high resolution

(18-days for July 9 - 27, 2006)

Altdorf (Reuss valley)



Lodrino (Leventina)



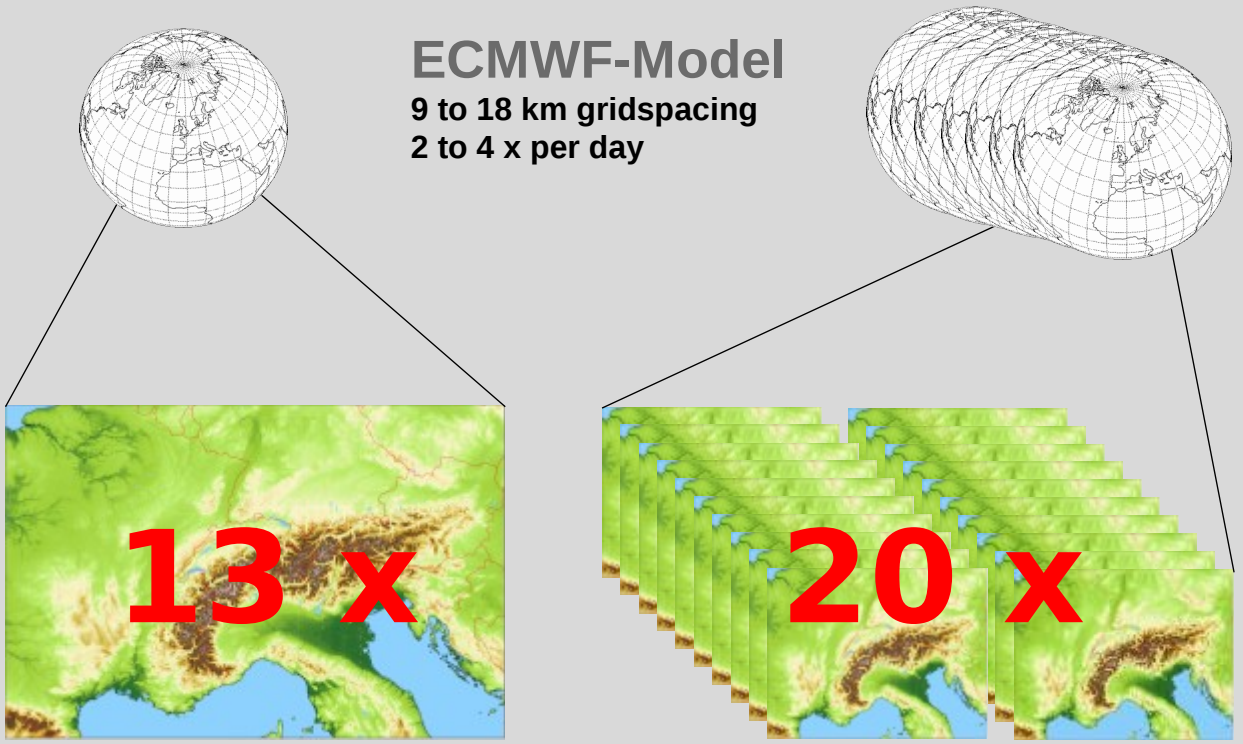
Observation Average wind speed (—) and direction (◊)

COSMO-2 **COSMO-1**

Courtesy of Jürg Schmidli, ETH

Computational cost = 40 x

(relative to current operational system)



Production with COSMO @ CSCS

Cray XE6 (Albis/Lema)
MeteoSwiss operational system
Since ~4 years

Next-generation system
Accounting for Moore's law (factor 4)

Images: CSCS

Co-design: Approach

- Design **software**, **workflow** and **hardware** with the following principles
  - Portability to other users (and hardware)
  - Achieve time-to-solution
  - Optimize energy (and space) requirements

- **Collaborative effort** between
  - MeteoSwiss, C2SM/ETH, CSCS for software since 2010
  - Cray and NVIDIA for new machine since 2013
  - Domain scientists and computer scientists

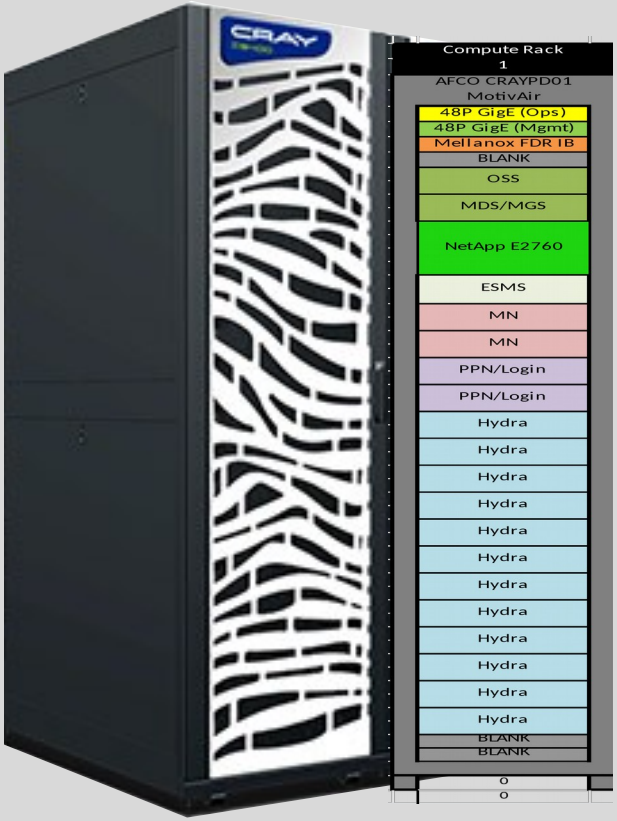
- Additional funding from the HPCN Strategy (HP2C, PASC)

Images: CSCS

New MeteoSwiss HPC system

Piz Kesch (Cray CS Storm)

- Installed at CSCS in July 2015
- Already Operational
- Hybrid system with a mixture of CPUs and GPUs
- “Fat” compute nodes with 2 Intel Xeon E5 2690 (Haswell) and 8 Tesla K80 (each with 2 GK210)
- Only 12 out of 22 possible compute nodes
- Fully redundant (failover for research and development)



Results

	Piz Dora	Piz Kesch	Factor
Sockets at required time-to-solution	~16 CPUs	~7 GPUs	**2.4 x**
Energy per member	6.19 kWh	2.06 kWh	**3.0 x**

.

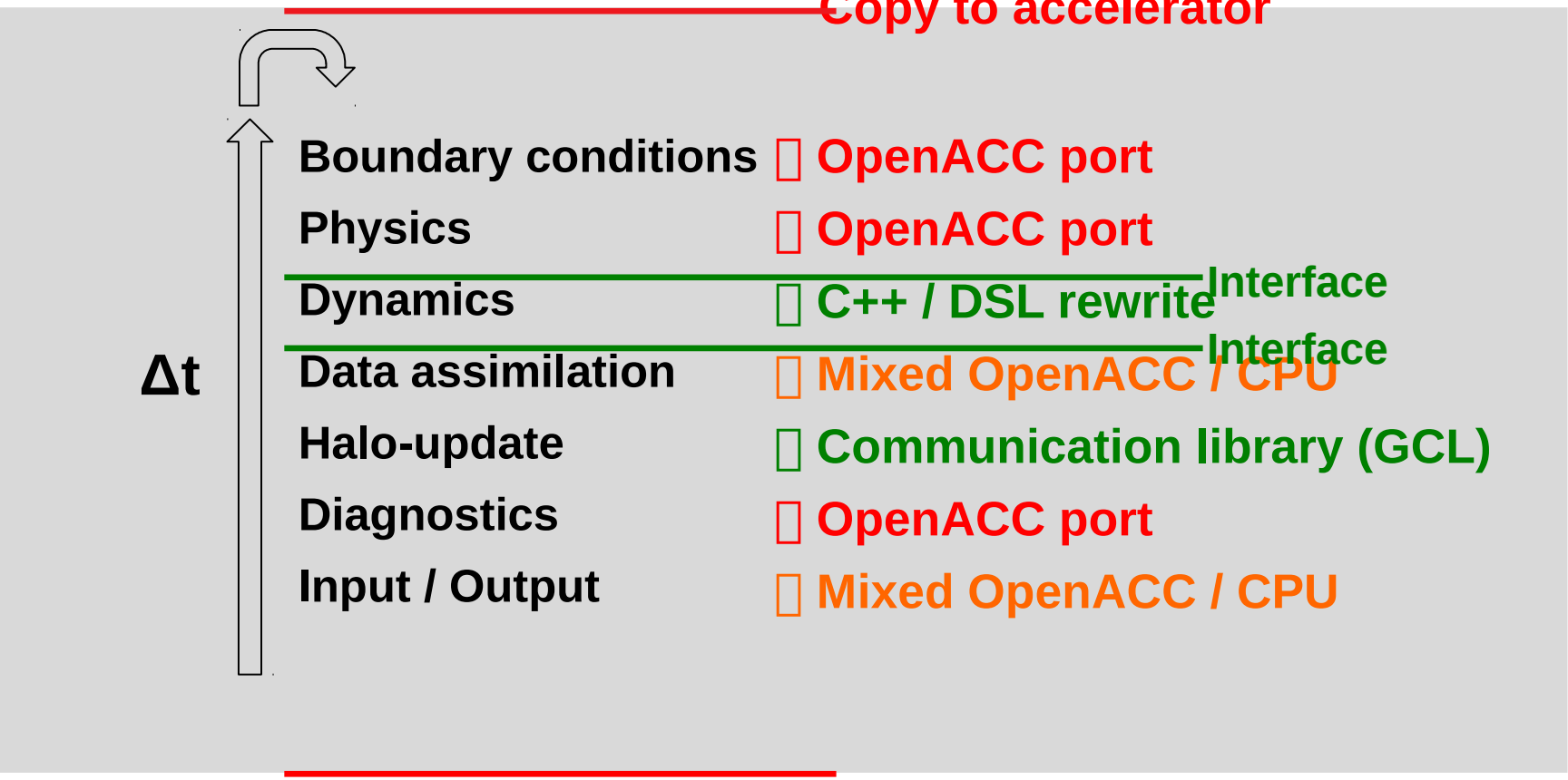
Results Relative to „Old“ Code

(„Old“ = no C++ dycore, double precision)

	Piz Dora	Piz Kesch	Factor
Sockets at required time-to-solution	~26 CPUs	~7 GPUs	**3.7 x**
Energy per member	10.0 kWh	2.06 kWh	**4.8 x**

Co-design: Software Technologies for Portable Models

- OpenACC
- C++ DSL for PDEs
- GCL for halo exchange communications (MPI based)



C++ DSL For Portable and Performance Portable Models?

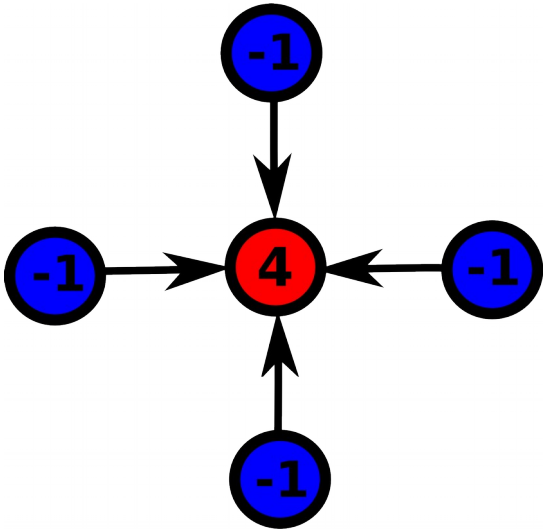
Separation of Concerns:

- Abstract hardware dependent code, underlying programming model from weather model.

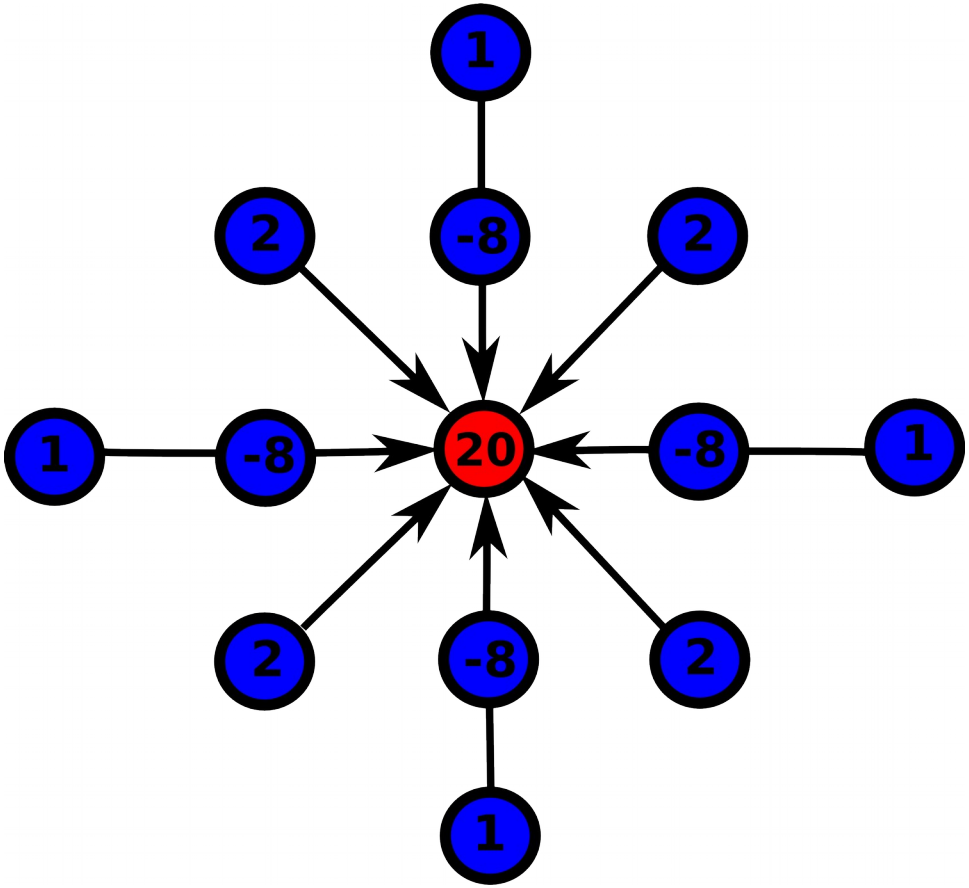
Main focus on dynamical cores: composition of complex stencils

Separation of Concerns

Laplace: $\nabla^2 u$



4th order horizontal diffusion: $-\alpha\nabla^2(\nabla^2 u)$



Naïve Implementation

```
for(int i=0; i < isize; ++i) {
    for(int j=0; j < jsize; ++j) {
        for(int k=0; k < ksize; ++k) {
            lap(i,j,k) = 4*u(i,j,k) - u(i+1,j,k) - u(i-1,j,k) -
                u(i,j+1,k) - u(i,j-1,k);
} } }
for(int i=0; i < isize; ++i) {
    for(int j=0; j < jsize; ++j) {
        for(int k=0; k < ksize; ++k) {
            u_new(i,j,k) = 4*lap(i,j,k) - lap(i+1,j,k) - lap(i-1,j,k) -
                lap(i,j+1,k) - lap(i,j-1,k);
} } }
```

- Readable (close to numerical formulation)
- Close to programming agnostic formulation

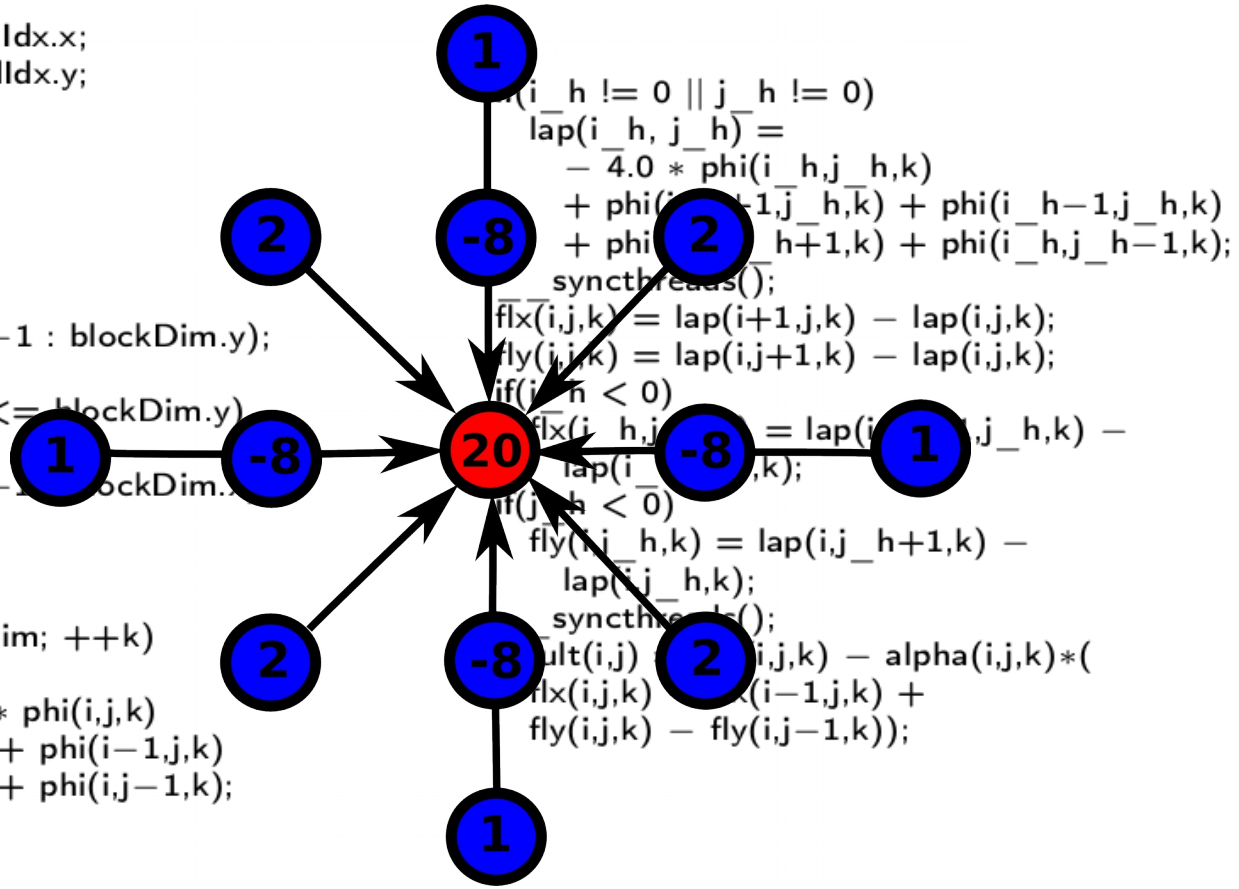
- Not Portable
- Not Parallel
- Not Optimized

Example kernel for 4th horizontal diffusion in CUDA

```
const int i = threadIdx.x;
const int j = threadIdx.y;
int i_h = 0;
int j_h = 0;

if(j < 2)
{
  i_h = i;
  j_h = (j==0 ? -1 : blockDim.y);
}
else if(j < 4 && i <= blockDim.y)
{
  i_h = (j==2 ? -1 : blockDim.x);
  j_h = i;
}

for(int k=0; k < kdim; ++k)
{
  lap(i,j) = - 4.0 * phi(i,j,k)
    + phi(i+1,j,k) + phi(i-1,j,k)
    + phi(i,j+1,k) + phi(i,j-1,k);

if(i_h != 0 || j_h != 0)
  lap(i_h, j_h) =
    - 4.0 * phi(i_h,j_h,k)
    + phi(i_h+1,j_h,k) + phi(i_h-1,j_h,k)
    + phi(i_h,j_h+1,k) + phi(i_h,j_h-1,k);
__syncthreads();
flx(i,j,k) = lap(i+1,j,k) - lap(i,j,k);
fly(i,j,k) = lap(i,j+1,k) - lap(i,j,k);
if(i_h < 0)
  flx(i_h,j_h,k) = lap(i_h+1,j_h,k) -
    lap(i_h,j_h,k);
if(j_h < 0)
  fly(i,j_h,k) = lap(i,j_h+1,k) -
    lap(i,j_h,k);
__syncthreads();
result(i,j) = phi(i,j,k) - alpha(i,j,k)*(
  flx(i,j,k) - flx(i-1,j,k) +
  fly(i,j,k) - fly(i,j-1,k));
```

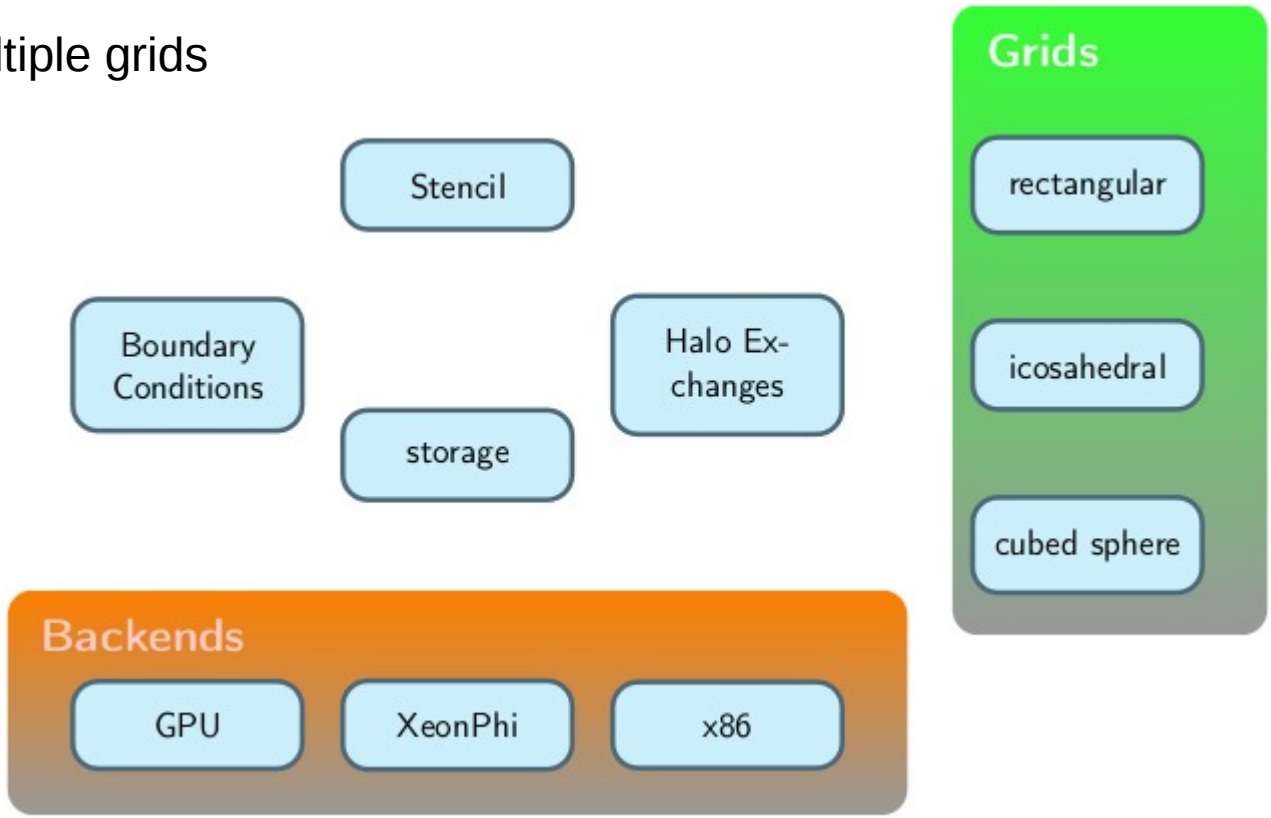
Example kernel for 4th horizontal diffusion in CUDA

```
const int i = threadIdx.x;
const int j = threadIdx.y;
int i_h = 0;
int j_h = 0;

if(j < 2)
{
  i_h = i;
  j_h = (j==0 ? -1 : blockDim.y);
}
else if(j < 4 && i <= blockDim.y)
{
  i_h = (j==2 ? -1 : blockDim.x);
  j_h = i;
}

for(int k=0; k < kdim; ++k)
{
  lap(i,j) = - 4.0 * phi(i,j,k)
    + phi(i+1,j,k) + phi(i-1,j,k)
    + phi(i,j+1,k) + phi(i,j-1,k);
```



GridTools

- Set of grid tools, including DSL for stencil codes, for solving PDEs on
  multiple grids



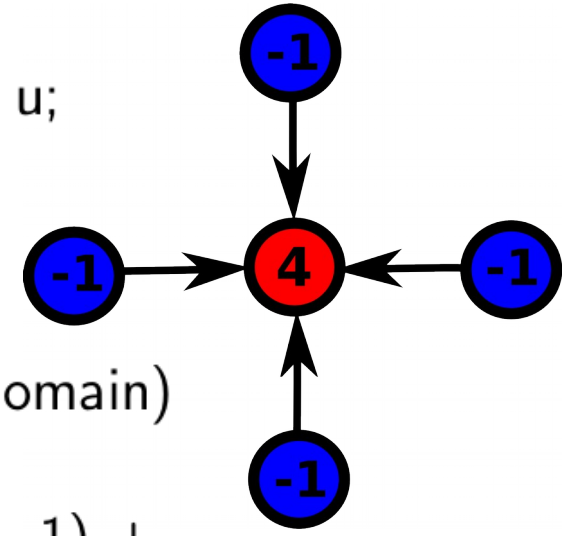
- Provides separation of concerns: Separates model and algorithm from hardware specific implementation and optimization
- Supports multiple hardware and grid backends.

Encoding Stencil Information in Types

```
struct Laplace
{
    typedef in_accessor<0, range<-1,1-1,1> > u;
    typedef out_accessor<1> lap;

    template<typename Evaluation>
    static void Do(Evaluation const& eval, full_domain)
    {
        eval(lap()) = eval(-4*u() + u(i+1) + u(i-1) +
            u(j+1) + u(j-1));
    }
};
```

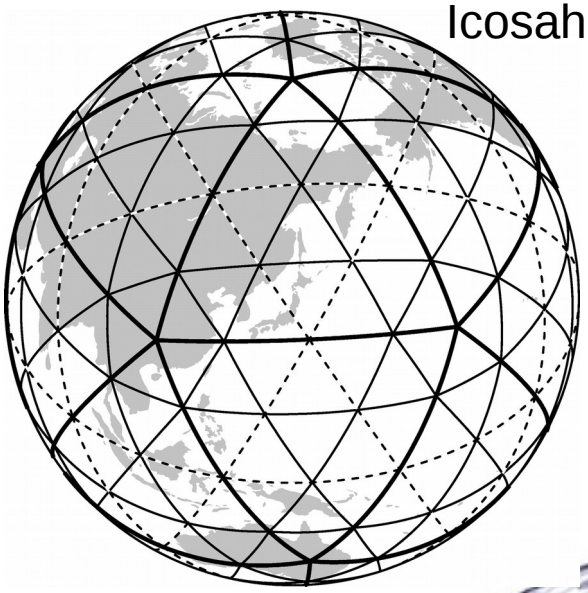
Encoding Stencil Information in Types

```
struct Laplace
{
    typedef in_accessor<0, range<−1,1−1,1> > u;
    typedef out_accessor<1> lap;

    template<typename Evaluation>
    static void Do(Evaluation const& eval, full_domain)
    {
        eval(lap()) = eval(−4∗u() + u(i+1) + u(i−1) +
            u(j+1) + u(j−1));
    }
};
```

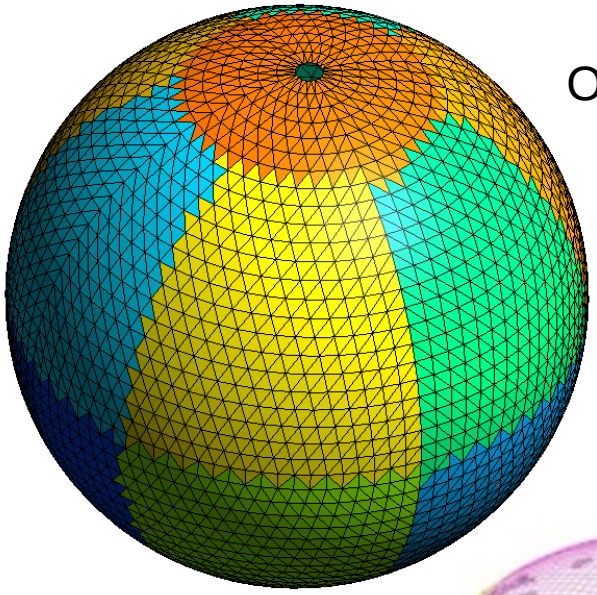


Support for Multiple Grids

Icosahedral

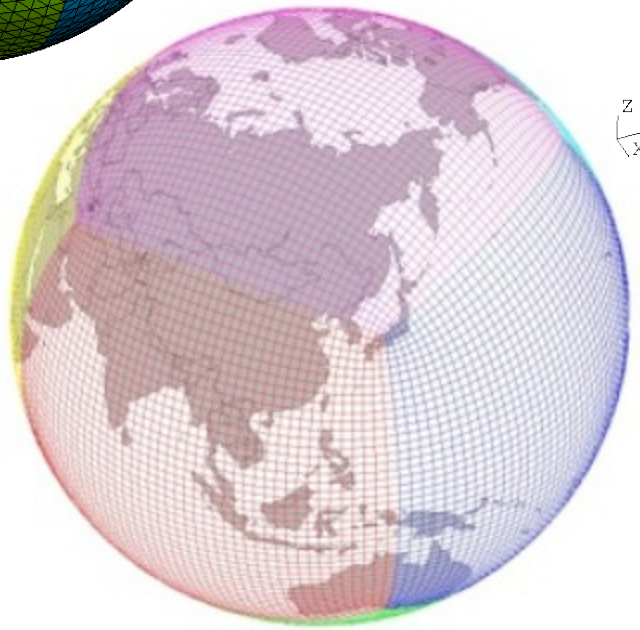


Ocahedral

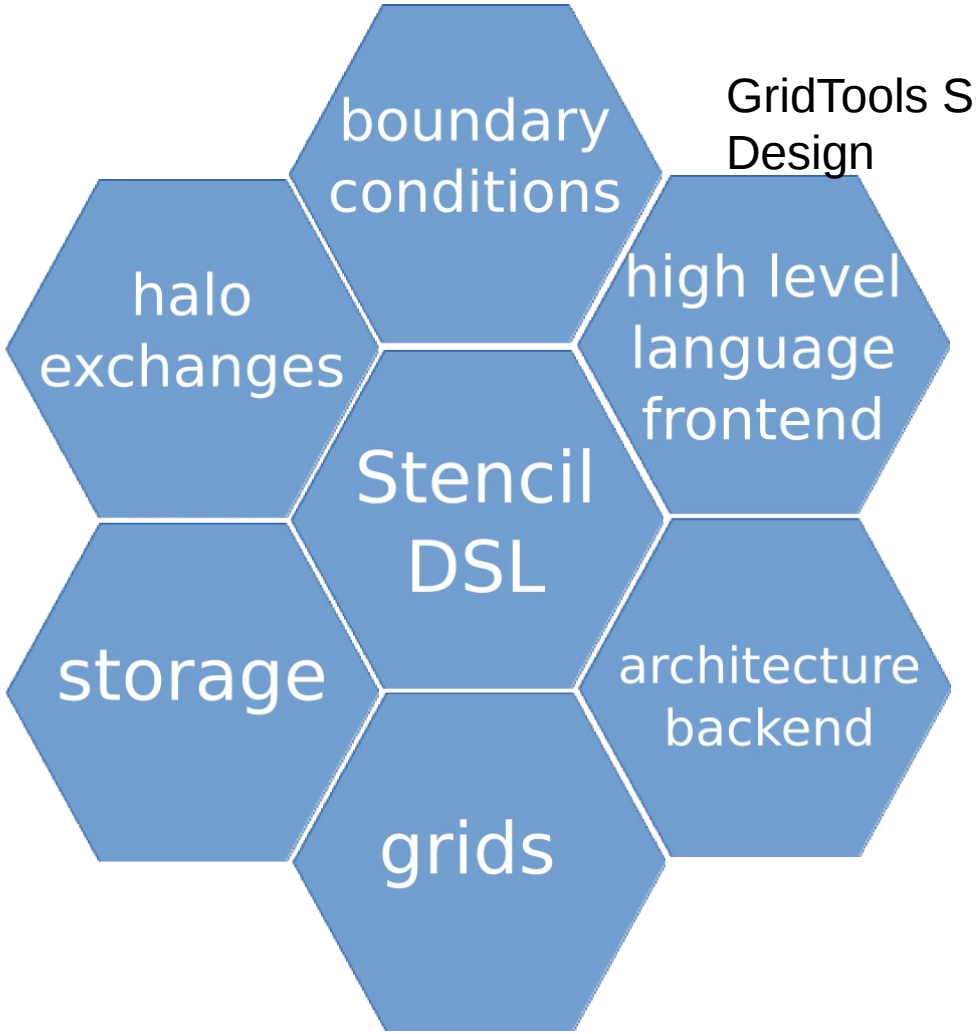


Dual-grid

Cube sphere

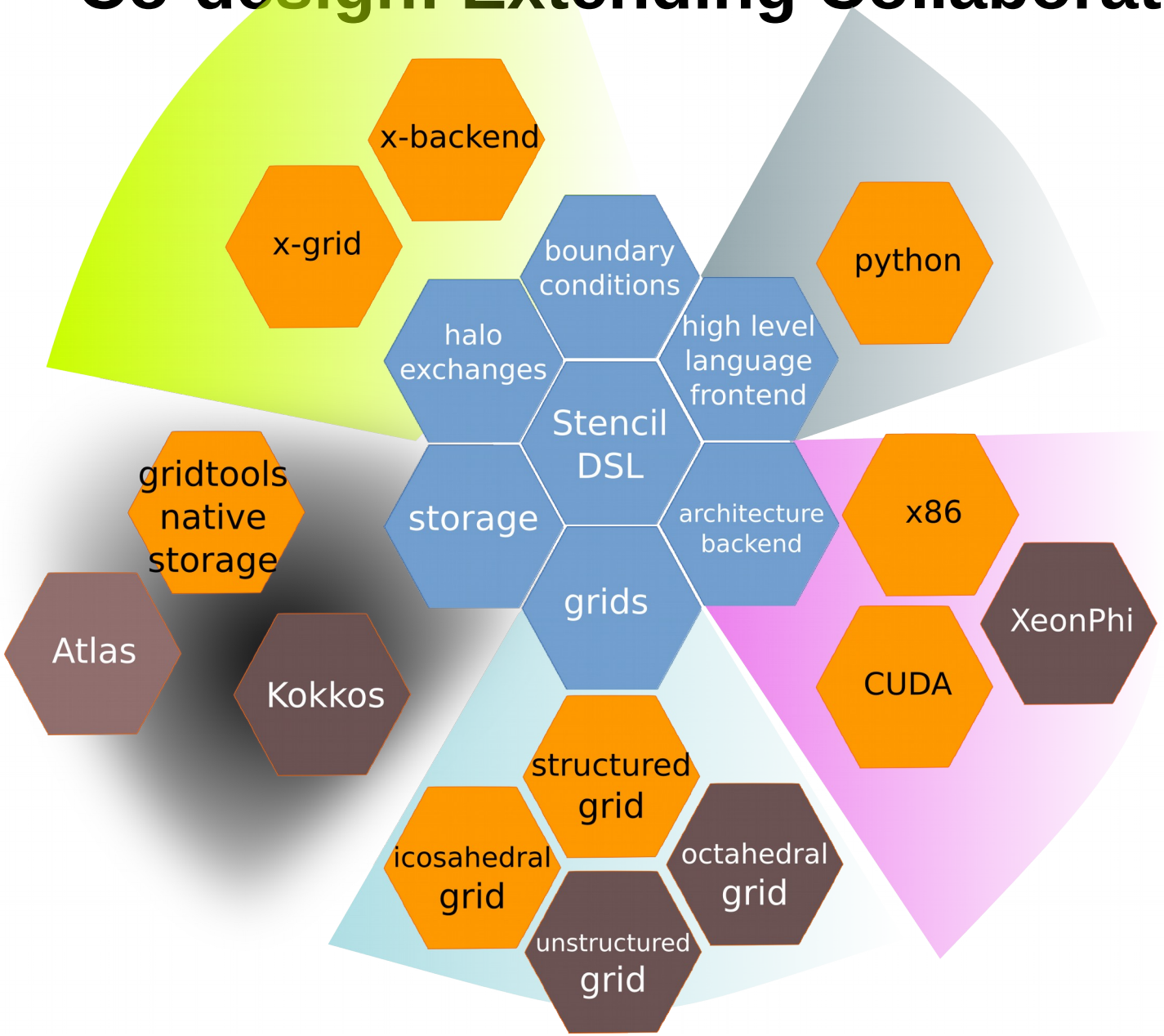


Co-design: Extending Collaborations

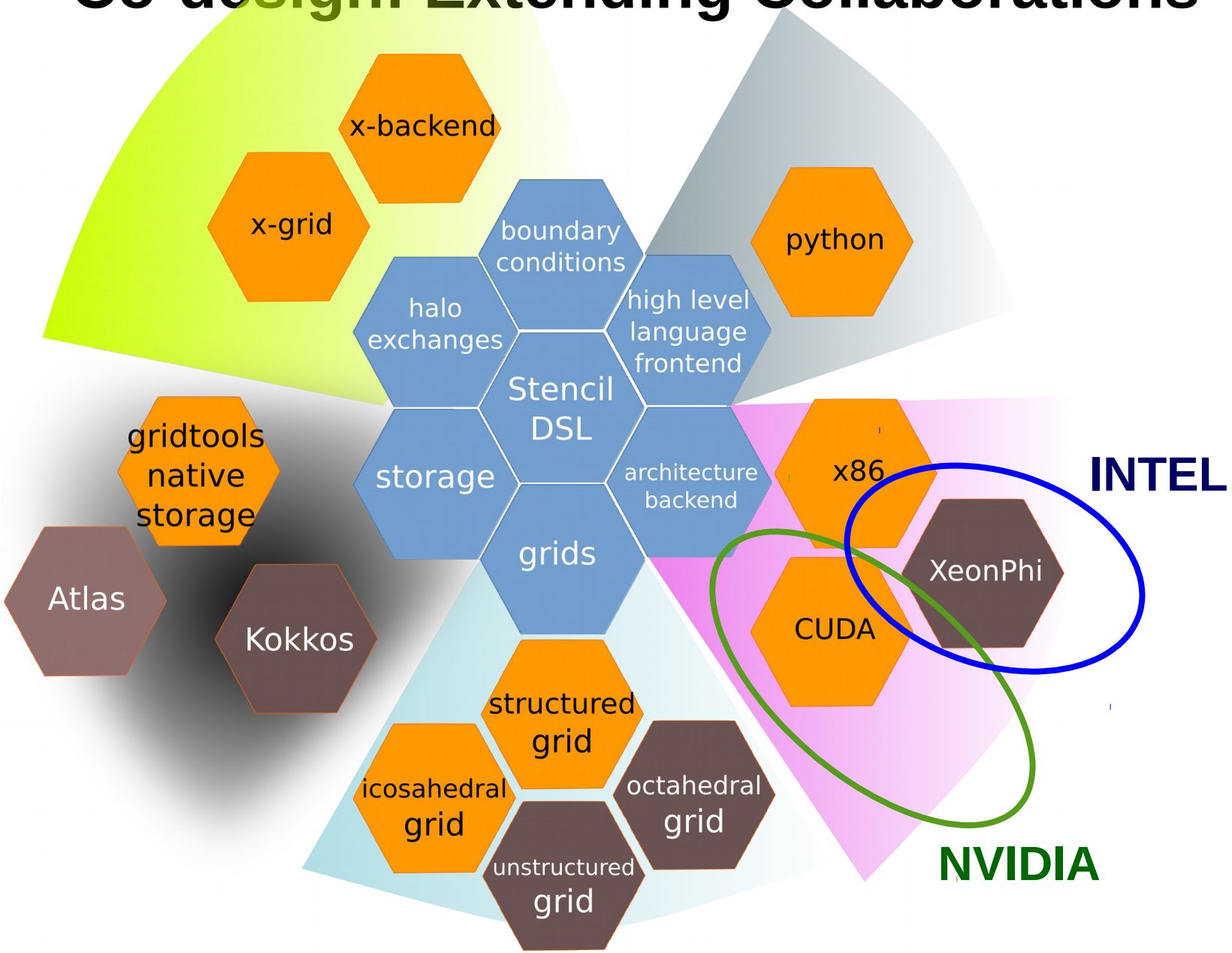


GridTools Software Design

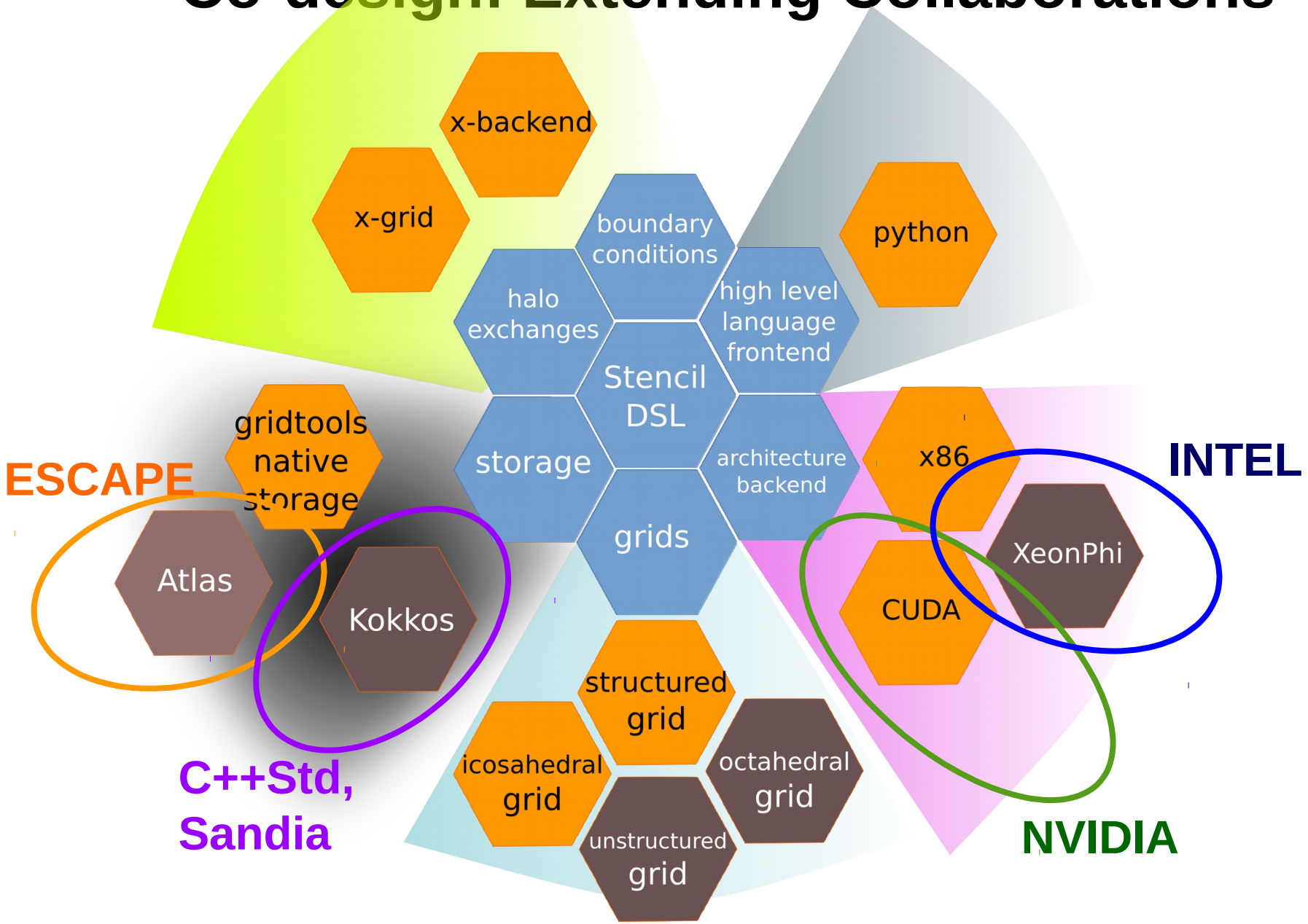
Co-design: Extending Collaborations



Co-design: Extending Collaborations



Co-design: Extending Collaborations



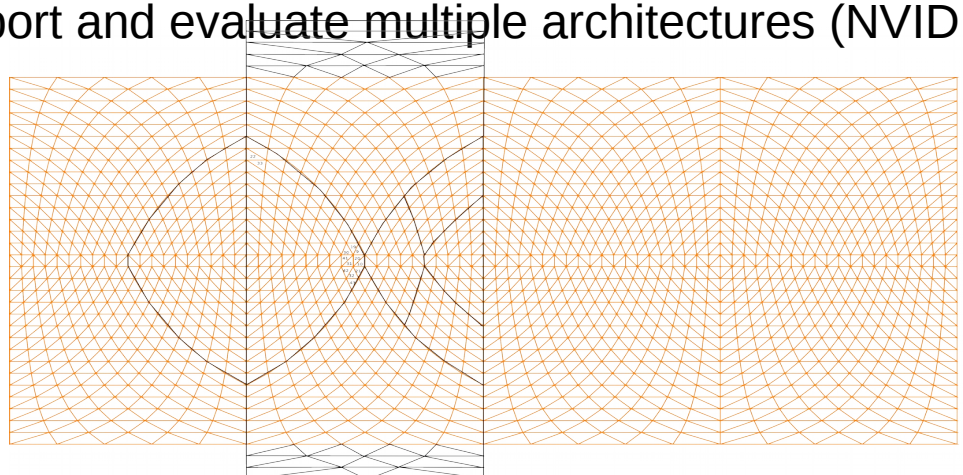
ESCAPE Project

ESCAPE (Energy-efficient Scalable Algorithms for Weather Prediction at Exascale)

Work on weather and climate “dwarfs”, explore programming models and adaptations to new computing architectures.

GridTools DSL for solvers on octahedral grids (structured and unstructured)

Support and evaluate multiple architectures (NVIDIA GPU, Xeon Phi)



Summary

- New forecasting system doubling resolution of deterministic forecast and introducing a convection permitting ensemble
- **Co-design**

(simultaneous code, hardware and workflow re-design in close collaboration with hardware & software vendors, model scientists, ...)

allowed MeteoSwiss to increase computational load by 40x within 4–5 years

- **Energy to solution is a factor 3x smaller** as compared to a “traditional” CPU-based system
- GridTools:
- Next generation DSL for PDEs: generalize, modular, grid & backend agnostic.

References

O. Fuhrer, C. Osuna, X. Lapillonne, T. Gysi, B. Cumming, M. Bianco, A. Arteaga, T. C. Schulthess, “**Towards a performance portable, architecture agnostic implementation strategy for weather and climate models**”, Supercomputing Frontiers and Innovations, vol. 1, no. 1 (2014), see http://superfri.org/

T. Gysi, C. Osuna, O. Fuhrer, M. Bianco and T. C. Schulthess, “**STELLA: A domain-specific tool for structure grid methods in weather and climate models**”, to be published in Proceedings of the International Conference on High-Performance Computing, Networking, Storage and Analysis, SC’15, New York, NY, USA (2015). ACM

Backups

Energy Measurement

- We use power clamp for comparison
- Measurements from PMDB and RUR were within 1% of clamp

Piz Dora (Cray XC40)

Power clamp

- (external measurement which measures wall consumption including AC/DC conversion, interconnect, but excluding blower)
- 1-2 nodes were down and could not be used (considered in computation)
- **PMDB** (1 Hz, per node)
- **RUR** (total per job)

Piz Kesch (Cray CS Storm)

Power clamp

- (external measurement which measures wall consumption including AC/DC conversion, interconnect, but excluding blower)
- Other components (mgmt nodes, extra service nodes, drives) powered down

„Managment summary“

Key ingredients

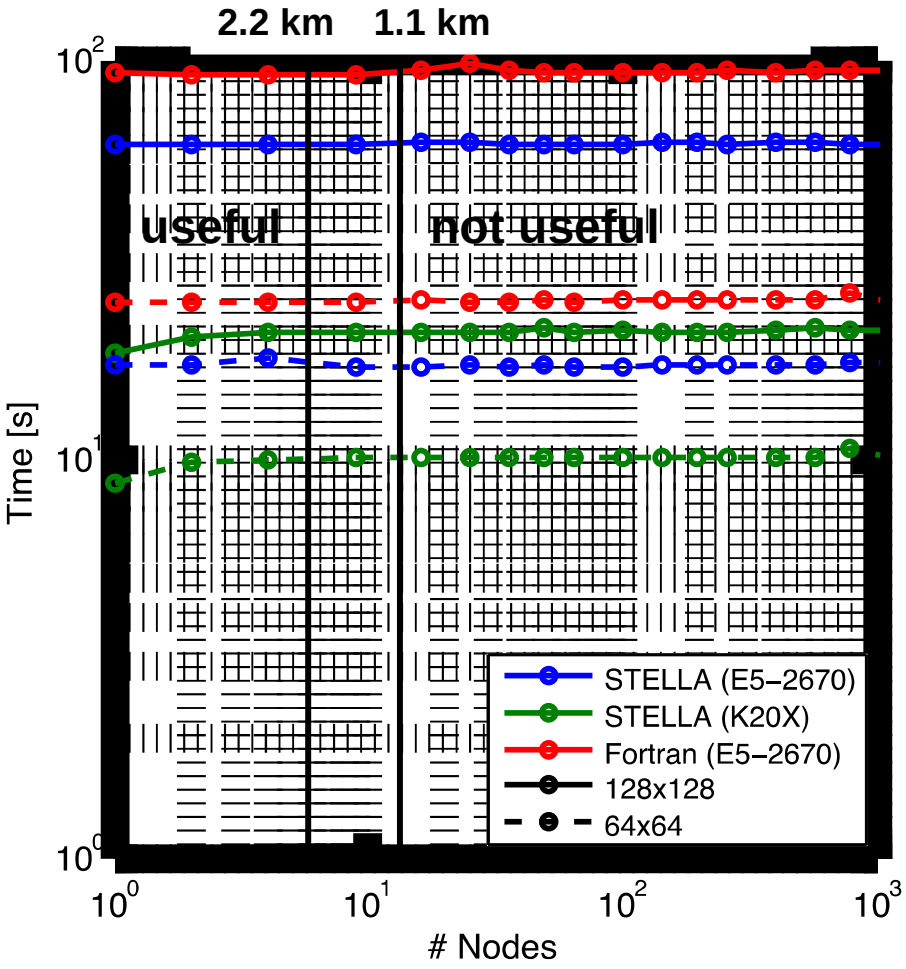
- Processor performance (Moore’s law) **~2.8 x**
- Port to accelerators (GPUs) **~2.3 x**
- Code improvement **~1.7 x**
- Increase utilization of system **~2.8 x**
- Increase in number of sockets **~1.3 x**
- Target system architecture to application

Note Separating hardware investments from software and workflow investments does not make sense!

Image: Cray

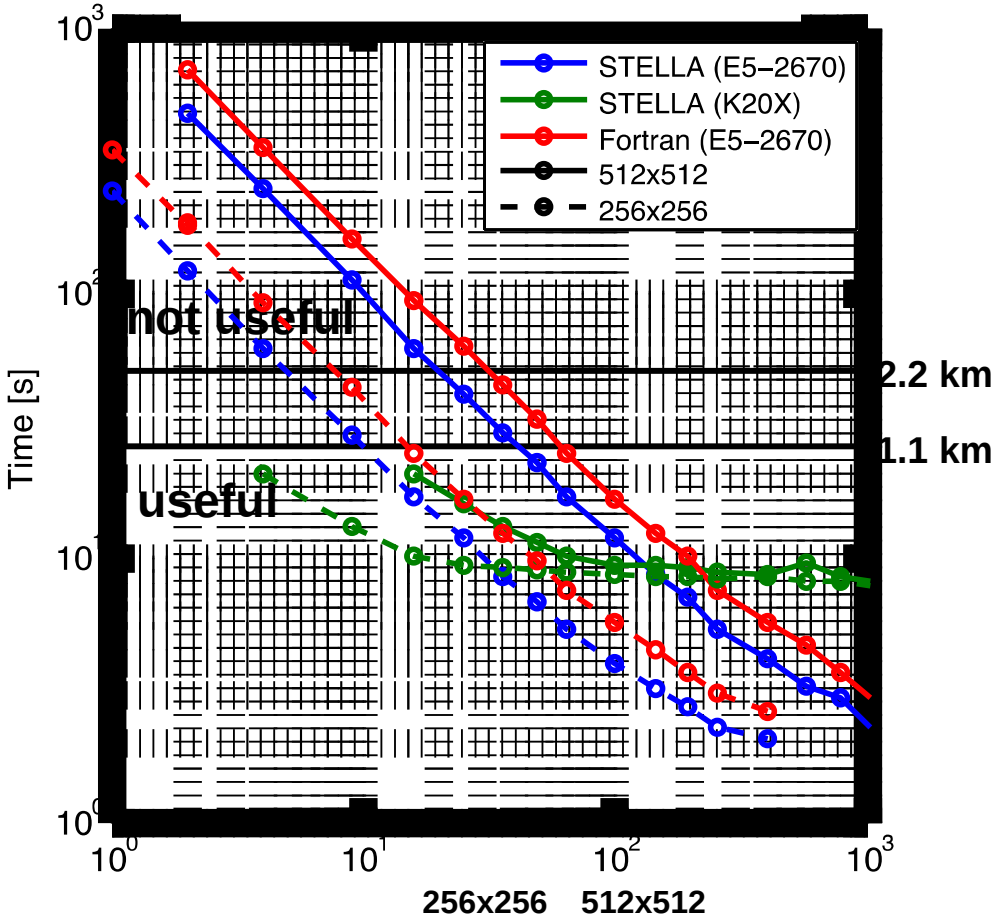
Scalability

Weak



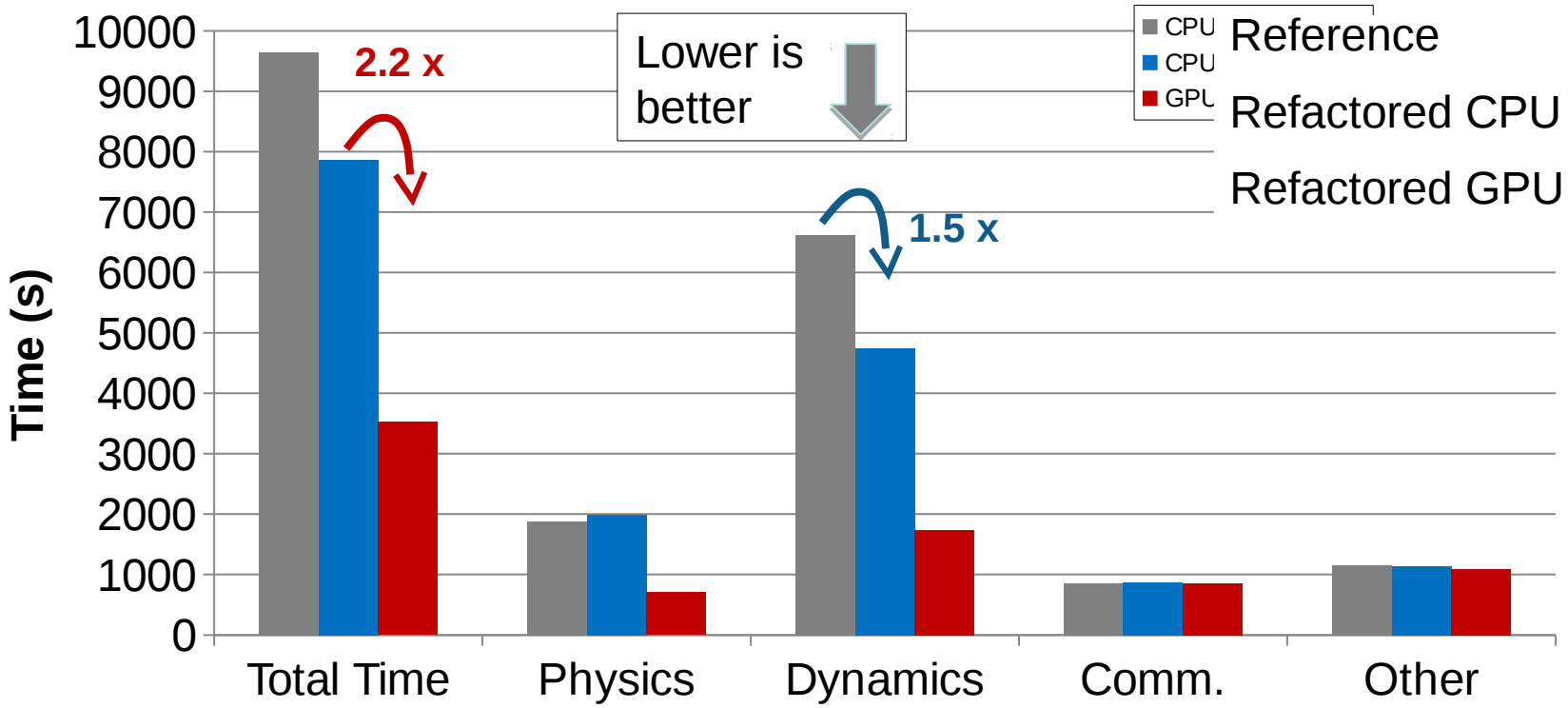
Gridpoints per node: 128x128x60 or 64x64x60

Strong



512x512x60 or 256x256x60 gridpoints total

Performance comparison



- 24h COSMO-1 forecast on 70 nodes of Piz Daint
- Refactoring effort in dynamical core (1.5 x)

Slim vs. fat nodes

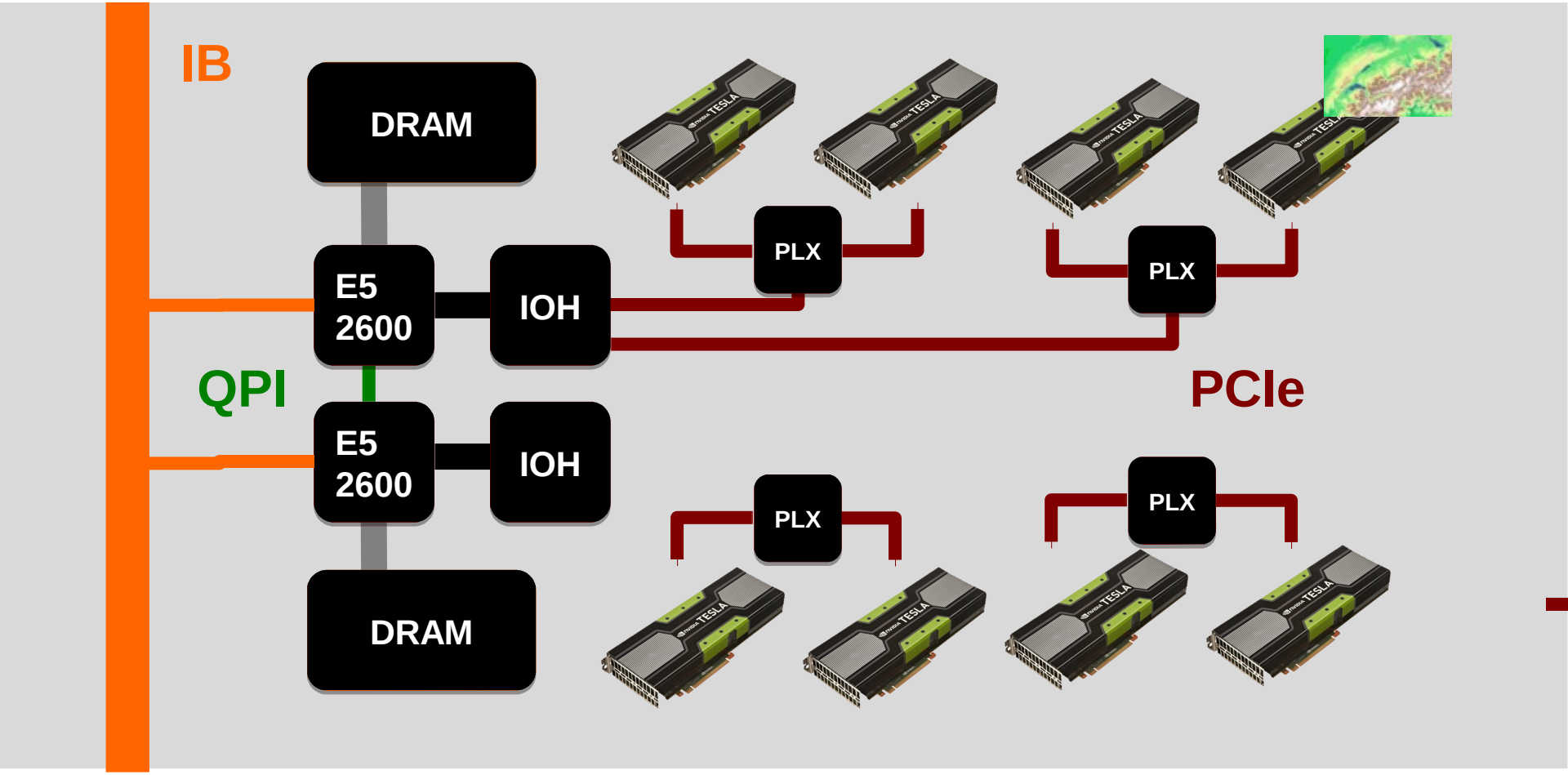
Piz Daint (Cay)
1 x Sandybridge
- 1 x NVIDIA Tesla K20x GPU
- 220 GB/s bandwidth

OPCODE (Tyan)
2 x Sandybridge
- 8 x NVIDIA Tesla K20 GPU
- 185 GB/s bandwidth

1402 s

1986 s with old version

G2G communication



G2G bandwidth

- Bi-directional BW between two GPUs

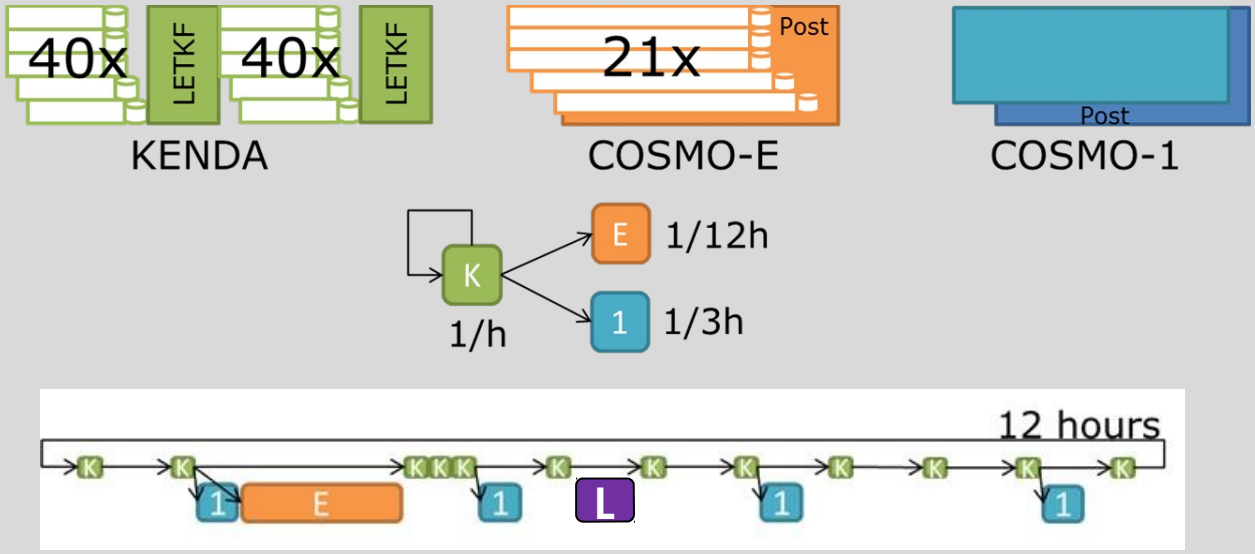
Same…	CUDA Memcopy	MPI	COSMO
PLX	13.1 GB/s	8.2 GB/s	**7.1 GB/s**
IOH	11.0 GB/s	7.4 GB/s	**6.9 GB/s**
Node	10.3 GB/s	4.2 GB/S	**4.1 GB/s**

- 2–8 GPUs)

2	3	4	5	6	7	**8**
7.0 GB/s	6.3 GB/s	6.5 GB/s	4.0 GB/s	4.5 GB/s	4.6 GB/s	**4.6 GB/s**

Workflow

- Pre- / post-processing software!
- Increase of machine utilization Error recovery and monitoring



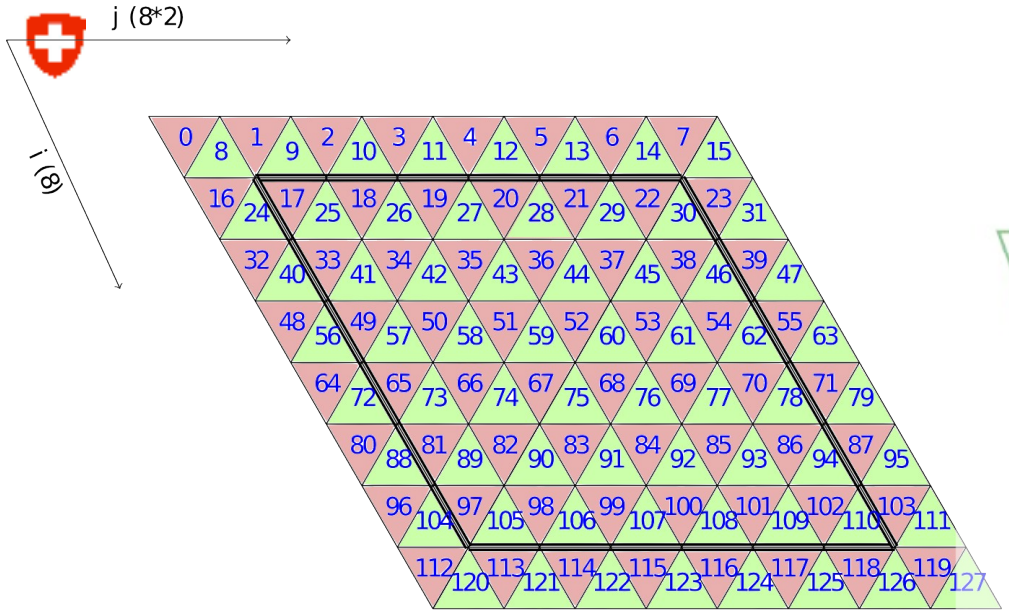
OpenACC vs. STELLA

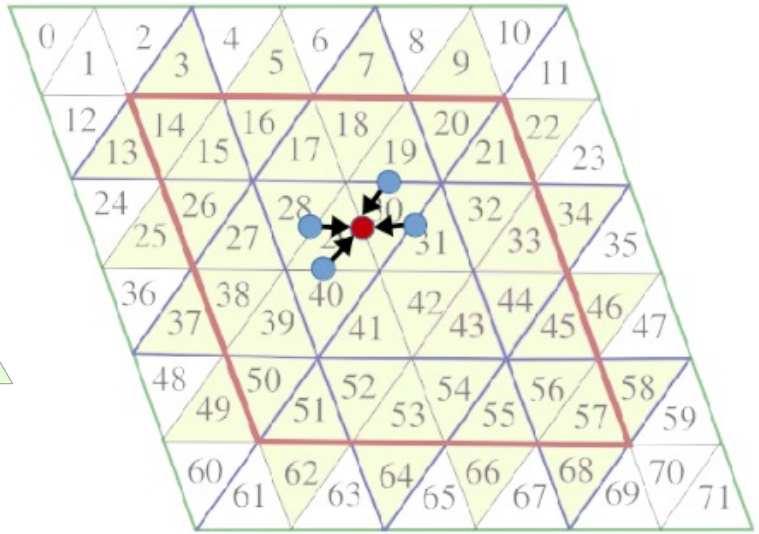
- Comparison using hor. diffusion and vert. advection

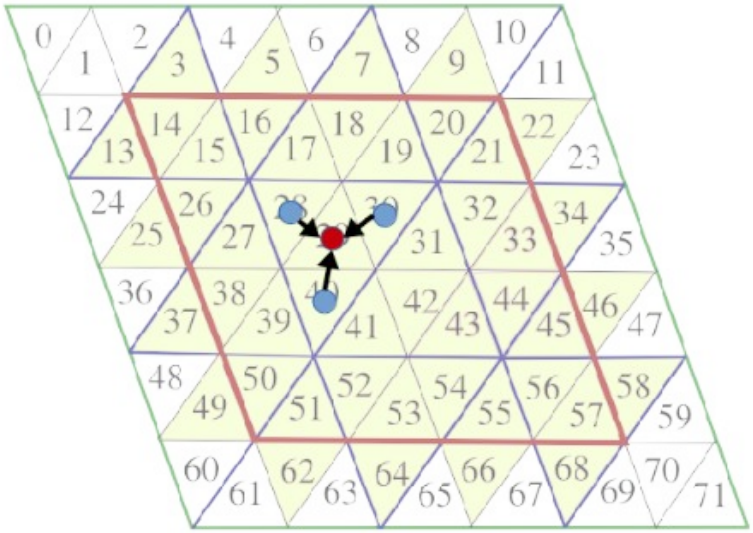
	runtime	occupancy	DRAM throughput read	DRAM throughput write	shared memory	register usage
non-blocked (naive)						
K20X	0.53 ms	0.266	>75.1 GB/s	>35.5 GB/s	0 B	47-53
K20	0.68 ms	0.285	>39.1 GB/s	>26.3 GB/s	0 B	37-44
blocked						
K20X	0.90 ms	0.283	13.9 GB/s	62.9 GB/s	0 B	73
K20	0.69 ms	0.591	12.7 GB/s	63.1 GB/s	4 B	46
shared						
K20	0.54 ms	0.600	15.9 GB/s	16.1 GB/s	4.272 KB	39
shared-3D						
K20	0.56 ms	0.670	15.4 GB/s	16.1 GB/s	4.272 KB	34
STELLA						
K20X	0.29 ms	0.90				
K20	0.35 ms	0.90				

Conclusions

- STELLA implementation is about 1.5 - 1.8x faster
- OpenACC code is portable, but not fully performance portable, many manual optimizations


j (8*2)
i (8)
0 8 1 9 2 10 3 11 4 12 5 13 6 14 7 15
16 24 17 25 18 26 19 27 20 28 21 29 22 30 23 31
32 40 33 41 34 42 35 43 36 44 37 45 38 46 39 47
48 56 49 57 50 58 51 59 52 60 53 61 54 62 55 63
64 72 65 73 66 74 67 75 68 76 69 77 70 78 71 79
80 88 81 89 82 90 83 91 84 92 85 93 86 94 87 95
96 104 97 105 98 106 99 107 100 108 101 109 102 110 103 111
112 120 113 121 114 122 115 123 116 124 117 125 118 126 119 127

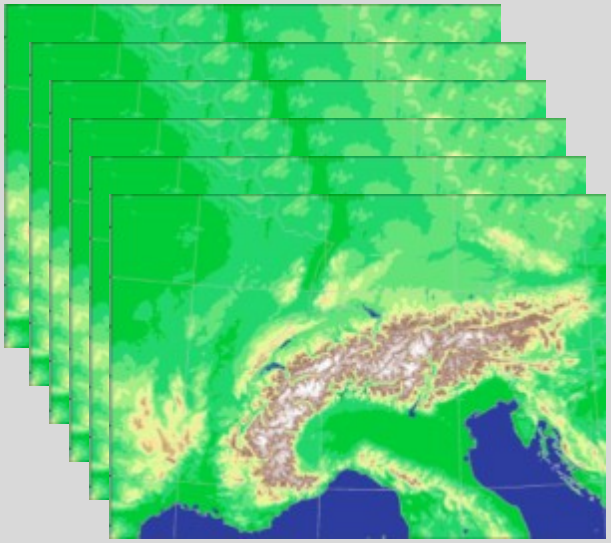

0 1 2 3 4 5 6 7 8 9 10 11
12 13 14 15 16 17 18 19 20 21 22 23
24 25 26 27 28 29 30 31 32 33 34 35
36 37 38 39 40 41 42 43 44 45 46 47
48 49 50 51 52 53 54 55 56 57 58 59
60 61 62 63 64 65 66 67 68 69 70 71


0 1 2 3 4 5 6 7 8 9 10 11
12 13 14 15 16 17 18 19 20 21 22 23
24 25 26 27 31 32 33 34 35
36 37 38 39 41 42 43 44 45 46 47
48 49 50 51 52 53 54 55 56 57 58 59
60 61 62 63 64 65 66 67 68 69 70 71

Benchmark

COSMO-E

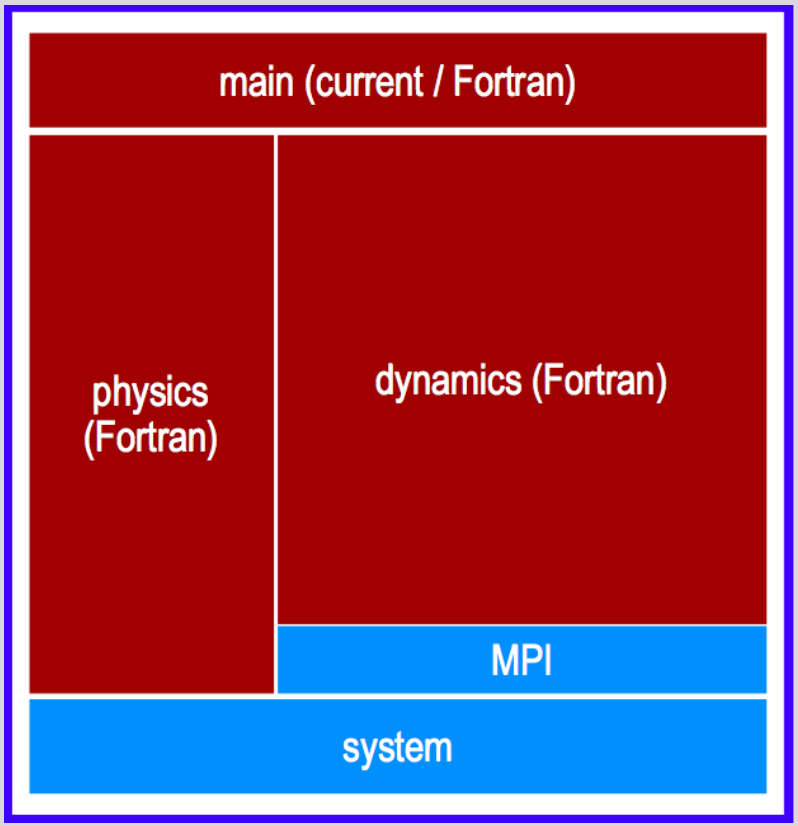
2.2 km gridspacing
582 x 390 x 60 gridpoints
120 h forecast

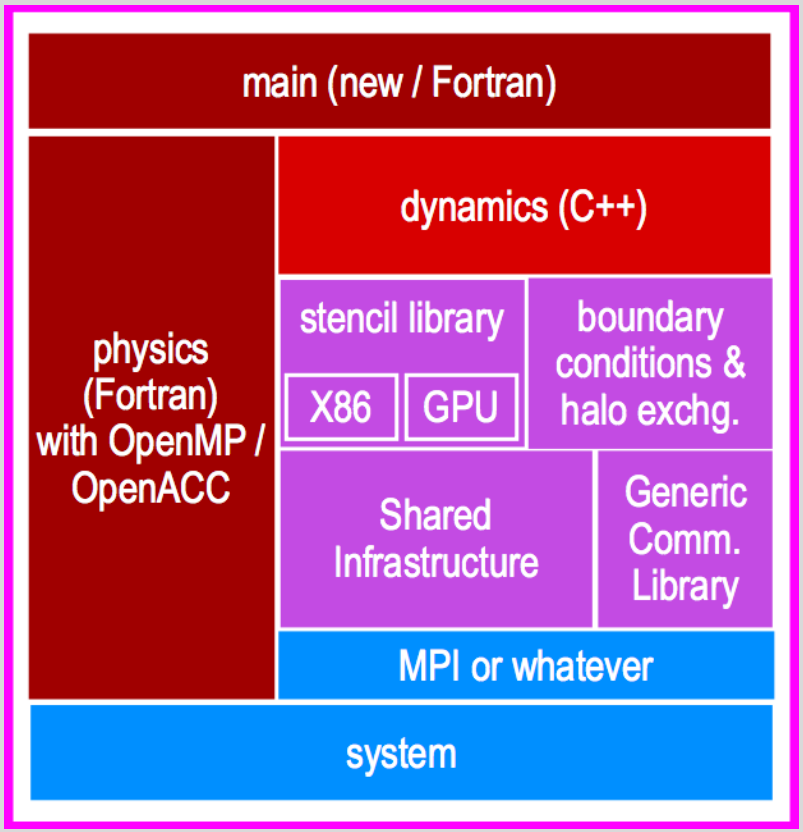


Details

- operational setup by MeteoSwiss
- Required time-to-solution = 2h
- (333 ms per timestep)
- Fill a full rack with members
- (keeping sockets per member constant)
- COSMO v5.0
- (with additions for GPU porting and C++ dynamical core)
- Single precision
- (both CPU and GPU not fully optimized)

Co-design: Software Design





adapted from Fuhrer et al. 2014