Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Eidgenössisches Departement des Innern EDI
**Bundesamt für Meteorologie und Klimatologie MeteoSchweiz**

# Numerical Weather Prediction on GPUs at MeteoSwiss

Xavier Lapillonne
09.06.2022

xavier.lapillonne@meteoswiss.ch

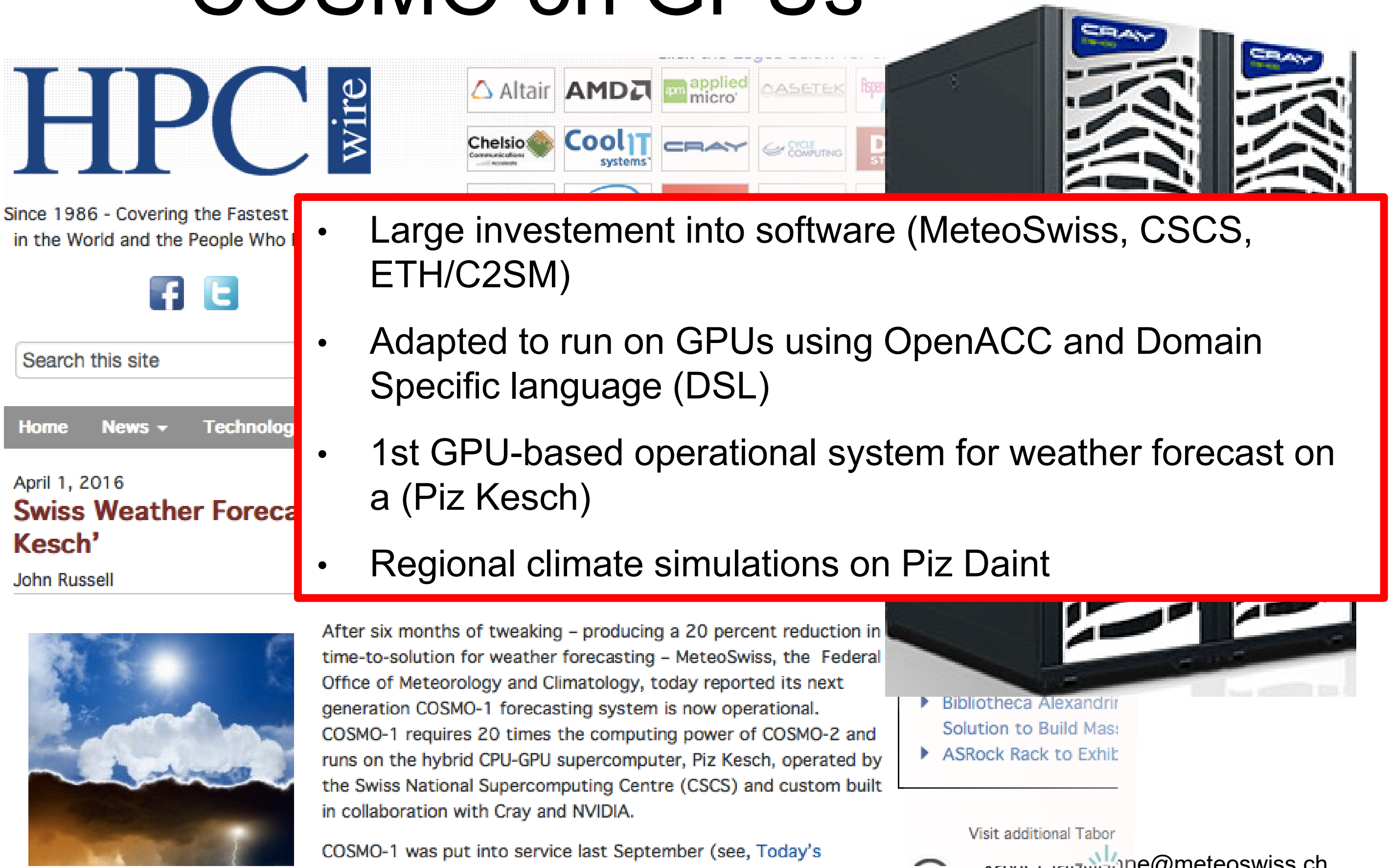
# Why using Domain Specific Language (DSL) in weather and climate ?

- DSL : computer language restricted to a particular domain
- We need performance to reach time to solution
- Separation of concern between domain and computer scientist
- Single source code for multiple target architectures
- Possible to write a new backend when a new technology emerged
- Allow aggressive optimization without degrading readability of user code
- Allow optimization across components – data centric optimization

xavier.lapillonne@meteoswiss.ch

# COSMO on GPUs

- Large investement into software (MeteoSwiss, CSCS, ETH/C2SM)
- Adapted to run on GPUs using OpenACC and Domain Specific language (DSL)
- 1st GPU-based operational system for weather forecast on a (Piz Kesch)
- Regional climate simulations on Piz Daint



Since 1986 - Covering the Fastest
in the World and the People Who

Search this site

Home News Technolog

April 1, 2016

**Swiss Weather Foreca**
**Kesch'**

John Russell

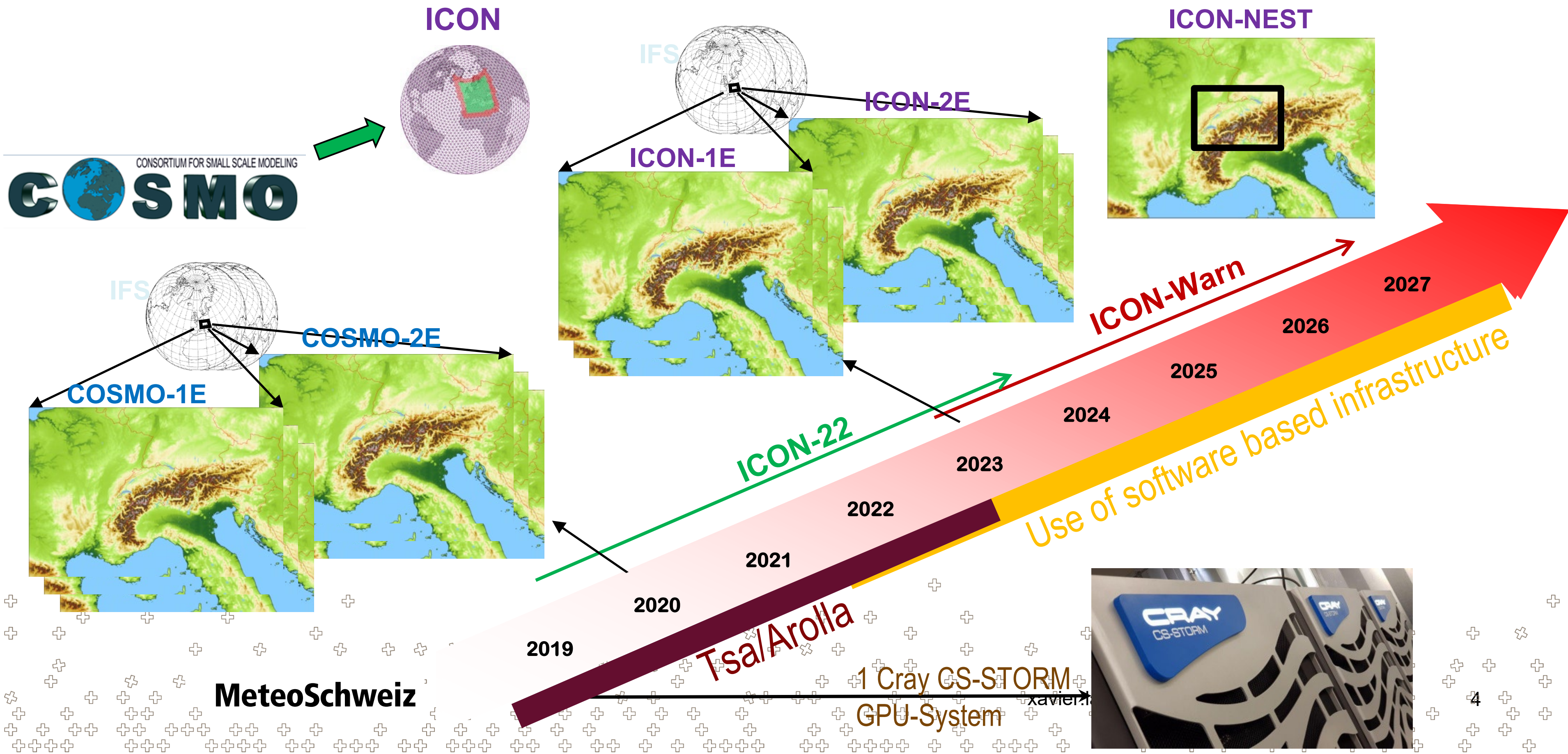
After six months of tweaking – producing a 20 percent reduction in time-to-solution for weather forecasting – MeteoSwiss, the Federal Office of Meteorology and Climatology, today reported its next generation COSMO-1 forecasting system is now operational. COSMO-1 requires 20 times the computing power of COSMO-2 and runs on the hybrid CPU-GPU supercomputer, Piz Kesch, operated by the Swiss National Supercomputing Centre (CSCS) and custom built in collaboration with Cray and NVIDIA.

COSMO-1 was put into service last September (see, Today's

- Bibliotheca Alexandrir
  Solution to Build Mas
- ASRock Rack to Exhit

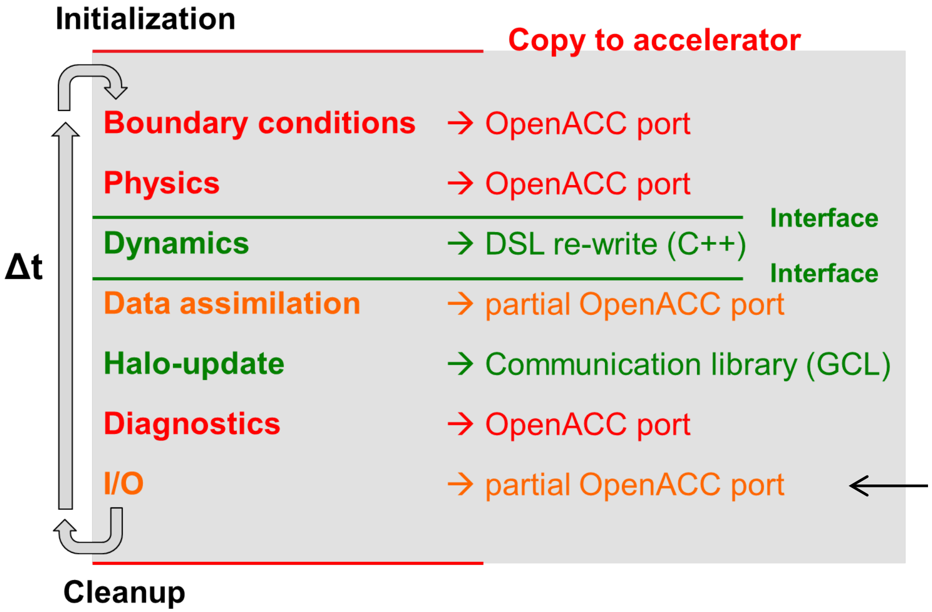
Visit additional Tabor

# Roadmap NWP Systems



# COSMO model on GPU

- Local area numerical weather prediction and climate model (cosmo-model.org), 350 KLOC F90 + MPI
- Full GPU port strategy : avoid GPU-CPU data transfer
- Approach: OpenACC compiler directives + Domain Specific Language (DSL) re-write
- 4-5x faster on GPU (socket to socket comparison)



**Initialization**

**Copy to accelerator**

$\Delta t$

**Boundary conditions** → OpenACC port

**Physics** → OpenACC port

**Interface**

**Dynamics** → DSL re-write (C++)

**Interface**

**Data assimilation** → partial OpenACC port

**Halo-update** → Communication library (GCL)

**Diagnostics** → OpenACC port

**I/O** → partial OpenACC port

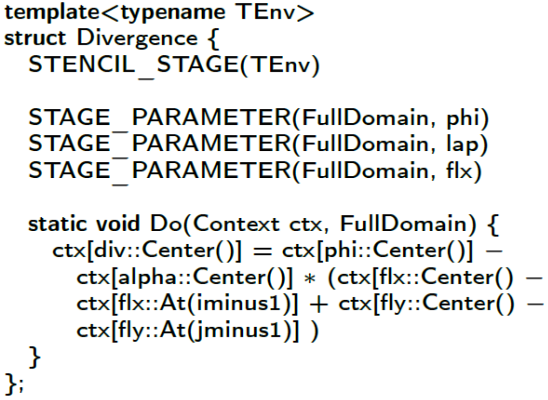
**Cleanup**

GPU-GPU communication

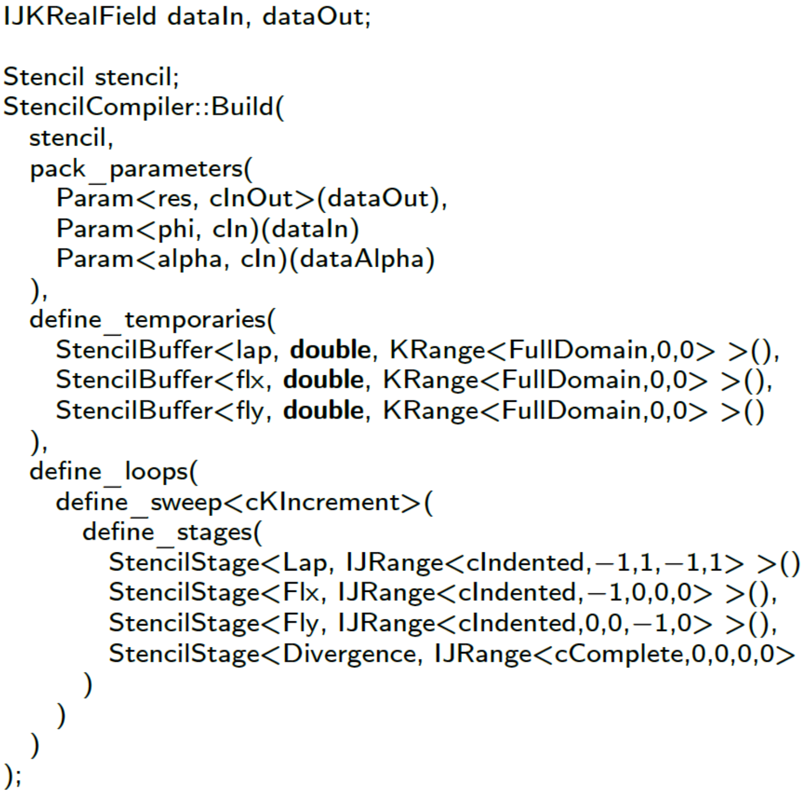
← Transfer to CPU
on I/O step only

# STELLA DSL for COSMO

First generation of DSL for weather models (C++, template metaprogramming)



```
template<typename TEnv>
struct Divergence {
  STENCIL_STAGE(TEnv)

  STAGE_PARAMETER(FullDomain, phi)
  STAGE_PARAMETER(FullDomain, lap)
  STAGE_PARAMETER(FullDomain, flx)

  static void Do(Context ctx, FullDomain) {
    ctx[div::Center()] = ctx[phi::Center()] −
      ctx[alpha::Center()] ∗ (ctx[flx::Center() −
      ctx[flx::At(iminus1)] + ctx[fly::Center() −
      ctx[fly::At(jminus1)] )
  }
};
```



```
IJKRealField dataIn, dataOut;

Stencil stencil;
StencilCompiler::Build(
  stencil,
  pack_parameters(
    Param<res, cInOut>(dataOut),
    Param<phi, cIn)(dataIn)
    Param<alpha, cIn)(dataAlpha)
  ),
  define_temporaries(
    StencilBuffer<lap, double, KRange<FullDomain,0,0> >(),
    StencilBuffer<flx, double, KRange<FullDomain,0,0> >(),
    StencilBuffer<fly, double, KRange<FullDomain,0,0> >()
  ),
  define_loops(
    define_sweep<cKIncrement>(
      define_stages(
        StencilStage<Lap, IJRange<cIndented,−1,1,−1,1> >()
        StencilStage<Flx, IJRange<cIndented,−1,0,0,0> >(),
        StencilStage<Fly, IJRange<cIndented,0,0,−1,0> >(),
        StencilStage<Divergence, IJRange<cComplete,0,0,0,0>
      )
    )
  )
);
```

# Lessons learned from COSMO port to GPU

- Stable operation at MeteoSwiss since 2016, 2 generations Cray Hybrid GPU systems

**OpenACC:**

+ Incremental insertion in existing code, good acceptance by domain scientist

- Some compiler bugs/issues, legacy code: no unittest, not always performance portable, only nvidia GPU (at this point), !$acc/omp conquer your code, !$acc is not comments but code.  Costly maintenance

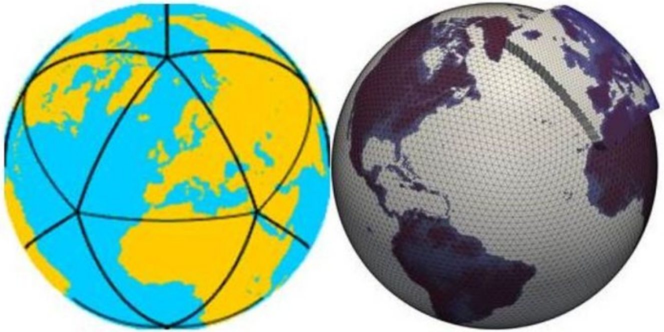
**DSL dycore:**

+ Separation of concerns, performance portable : tested on CPU, supports various hardware architecture (Nvidia, AMD GPU, Intel MIC, easier to test and maintain (unittest)

- Low acceptance, new syntax and black box, requires new know how, limited to DSL supported features, hard to debug (C++ Meta Programming)

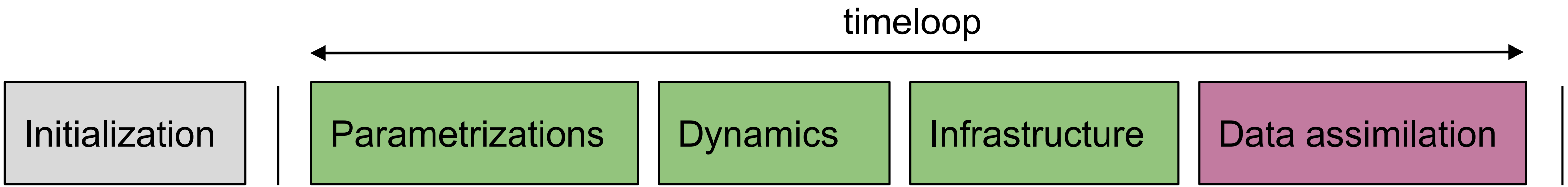
# ICON port to GPU

- ICON: Non-hydrostatic global and regional unified climate and numerical weather prediction model.
- ICON partners: DWD, MPI-M, DKRZ, KIT – ICON dev. Partners: COSMO, C2SM …
- Initial GPU port: OpenACC only, accepted by all partners, step by step integration in official version (MeteoSwiss, C2SM, CSCS, DWD, DKRZ, MPI-M)
- DSL implementation of components - research projects : ESCAPE, ESiWACE, EXCLAIM, WarmWorld



xavier.lapillonne@meteoswiss.ch

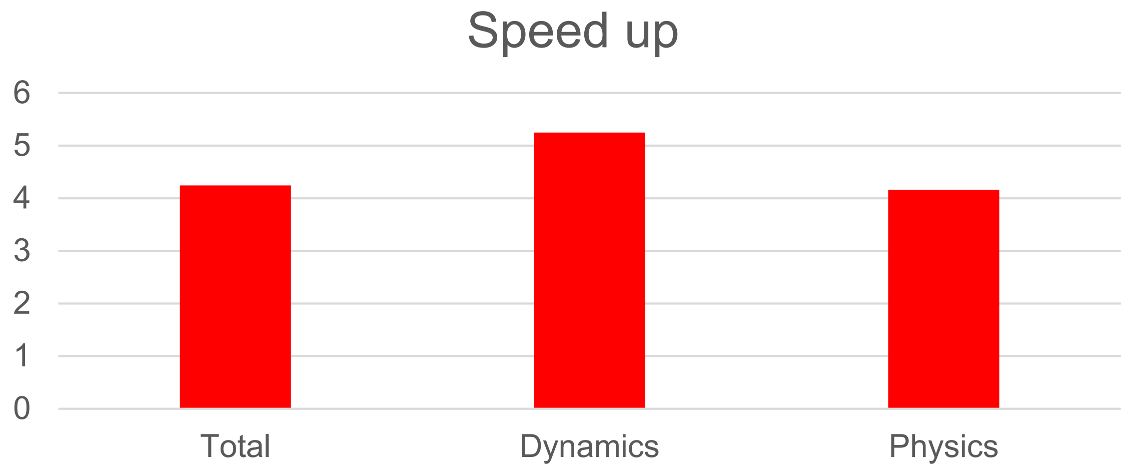
# Porting strategy ICON: OpenACC



- Avoid GPU-CPU transfer: all components of the time loop need to be ported to GPU
  - Exception: Data assimilation runs on CPU
- Design : Main parallelization along horizontal blocking (nproma - not at the block level like OpenMP for CPU)
  - Use large nproma when running on GPU (ideally 1 single block per GPU)
  - Compatible with COSMO parameterizations already ported to GPU
- Testing:
  - Individual components with Serialbox and PCAST (Nvidia compiler tool) during development
  - Comparing model output with tolerance (CPU/GPU not bit-identical)
- All components for NWP Regional application ported, optimization work ongoing.
- Port of components for the global setup ongoing

# CPU – GPU comparison (socket to socket)

Operational ICON-2 (2 km. Alps) 8 Nodes, 1h, P100 GPU vs Intel Xeon E5 12 cores (Piz Daint, CSCS, GPU node)



Overall improvement ca 4.2x, optimization ongoing.
Note : this is not enough for operational requirement at MeteoSwiss, slower compare to COSMO

xavier.lapillonne@meteoswiss.ch

# Porting and optimization challenges

## OpenACC optimizations

- GPU and CPU working asynchronously
  - Reduces launch overhead
- Bundling similar loop constructs into single GPU kernels
  - Improves cache reuse
  - Reduces launch overhead
- Compiler assisted / manual inlining of function calls
  - Required for complex (deep call-trees) GPU kernels
  - Enables optimizations above

## Conceptual challenges

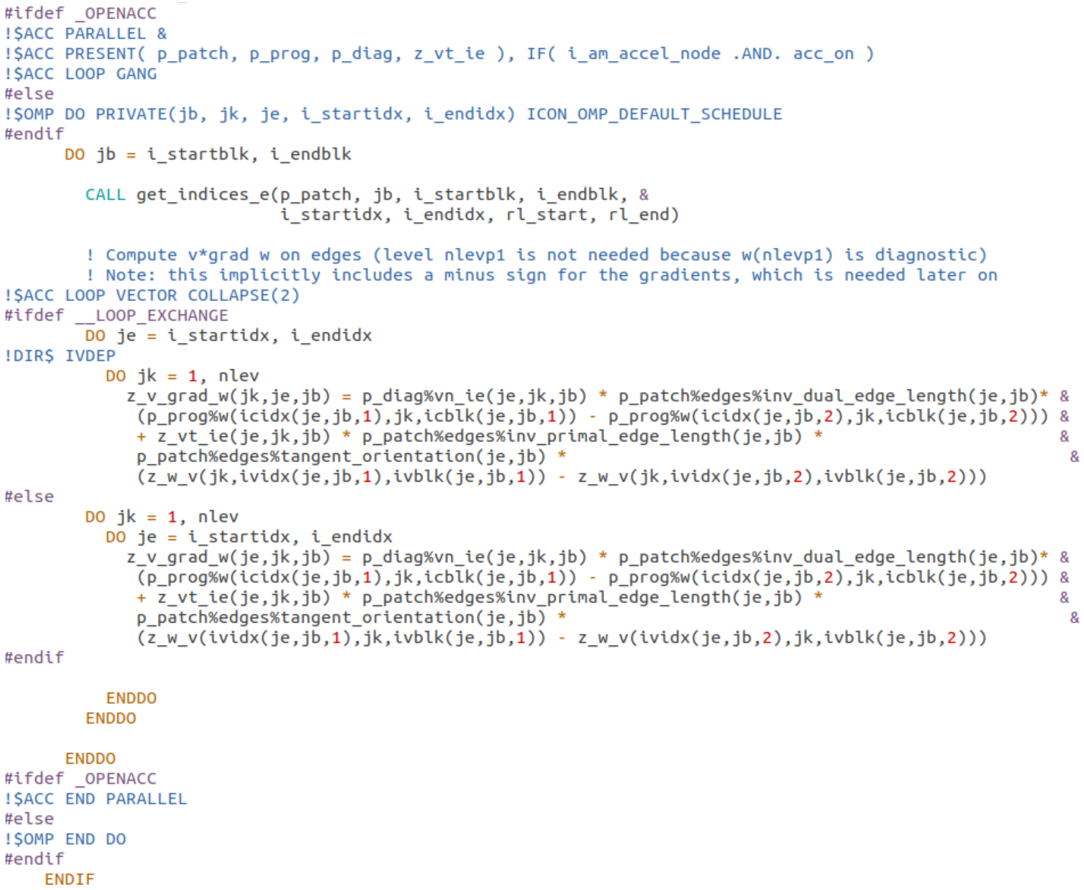
- Tiling for surface and turbulence
  - Implicitly introduces sub-blocking which leads to underutilized GPUs
- Physics initialization on CPU
  - Prohibitively slow because of unsuitable nproma and MPI settings for CPU
- Radiation sub-blocking
  - Radiation (ec-rad) has an additional dimension which can be parallelized Sub-blocking as a memory optimization
- Code management
  - Disruptive code changes are challenging
  - ecrad: juggling diverse interests

xavier.lapillonne@meteoswiss.ch

# GPU Port Technologies (OpenACC)

OpenACC+Fortran :

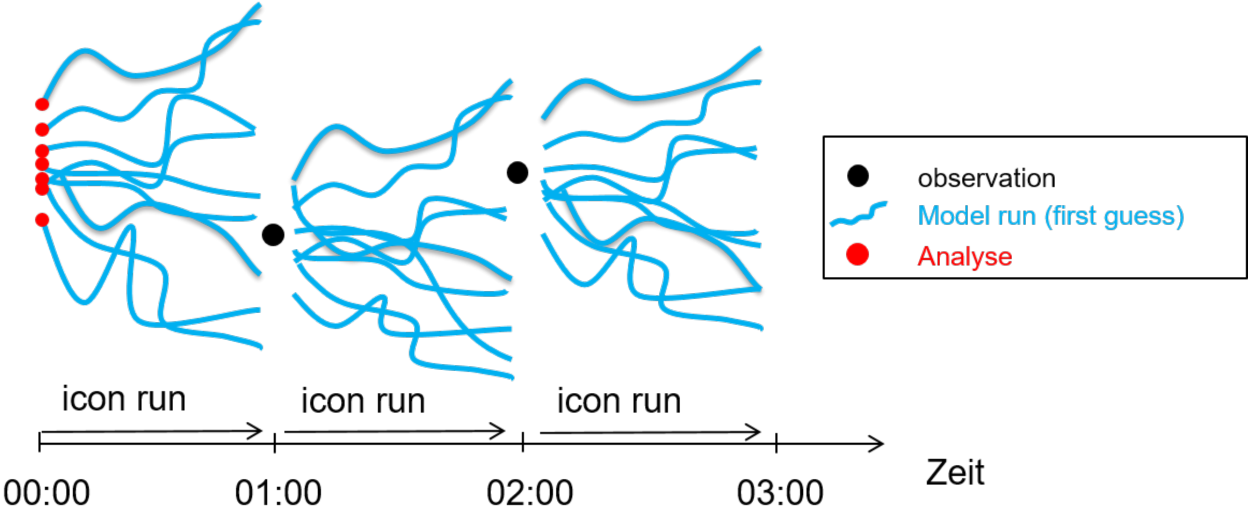
- Many ifdef
- Limited optimization margin
- Currently only works on Nvidia GPUs



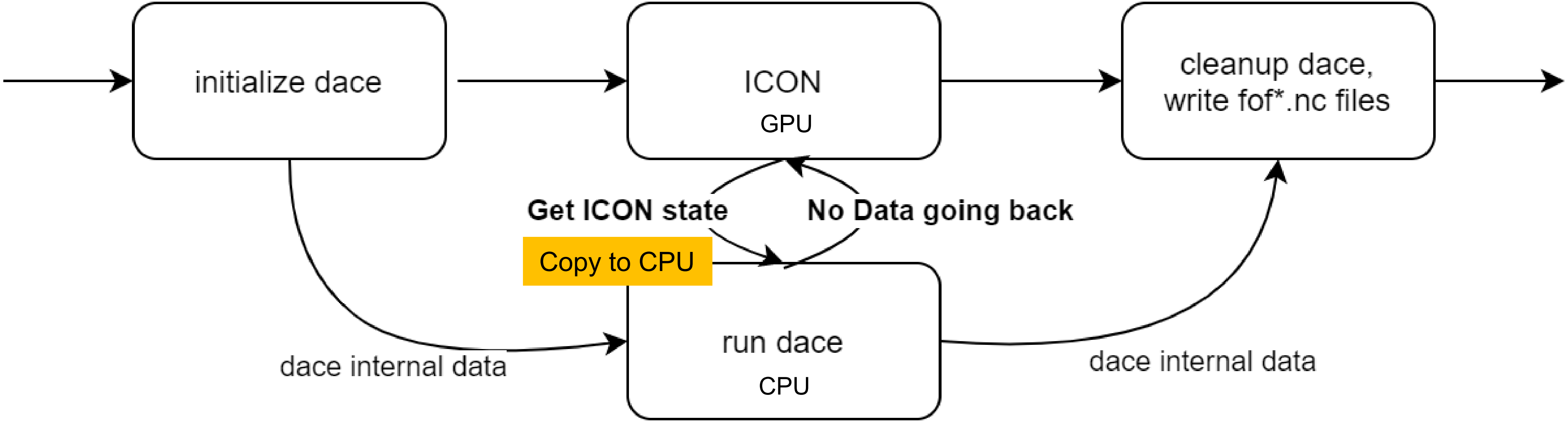
```
#ifdef _OPENACC
!$ACC PARALLEL &
!$ACC PRESENT( p_patch, p_prog, p_diag, z_vt_ie ), IF( i_am_accel_node .AND. acc_on )
!$ACC LOOP GANG
#else
!$OMP DO PRIVATE(jb, jk, je, i_startidx, i_endidx) ICON_OMP_DEFAULT_SCHEDULE
#endif
      DO jb = i_startblk, i_endblk

        CALL get_indices_e(p_patch, jb, i_startblk, i_endblk, &
                           i_startidx, i_endidx, rl_start, rl_end)

        ! Compute v*grad w on edges (level nlevp1 is not needed because w(nlevp1) is diagnostic)
        ! Note: this implicitly includes a minus sign for the gradients, which is needed later on
!$ACC LOOP VECTOR COLLAPSE(2)
#ifdef __LOOP_EXCHANGE
        DO je = i_startidx, i_endidx
!DIR$ IVDEP
          DO jk = 1, nlev
            z_v_grad_w(jk,je,jb) = p_diag%vn_ie(je,jk,jb) * p_patch%edges%inv_dual_edge_length(je,jb)* &
             (p_prog%w(icidx(je,jb,1),jk,icblk(je,jb,1)) - p_prog%w(icidx(je,jb,2),jk,icblk(je,jb,2))) &
             + z_vt_ie(je,jk,jb) * p_patch%edges%inv_primal_edge_length(je,jb) *                         &
             p_patch%edges%tangent_orientation(je,jb) *                                                   &
             (z_w_v(jk,ividx(je,jb,1),ivblk(je,jb,1)) - z_w_v(jk,ividx(je,jb,2),ivblk(je,jb,2)))
#else
        DO jk = 1, nlev
          DO je = i_startidx, i_endidx
            z_v_grad_w(je,jk,jb) = p_diag%vn_ie(je,jk,jb) * p_patch%edges%inv_dual_edge_length(je,jb)* &
             (p_prog%w(icidx(je,jb,1),jk,icblk(je,jb,1)) - p_prog%w(icidx(je,jb,2),jk,icblk(je,jb,2))) &
             + z_vt_ie(je,jk,jb) * p_patch%edges%inv_primal_edge_length(je,jb) *                         &
             p_patch%edges%tangent_orientation(je,jb) *                                                   &
             (z_w_v(ividx(je,jb,1),jk,ivblk(je,jb,1)) - z_w_v(ividx(je,jb,2),jk,ivblk(je,jb,2)))
#endif

          ENDDO
        ENDDO

      ENDDO
#ifdef _OPENACC
!$ACC END PARALLEL
#else
!$OMP END DO
#endif
    ENDIF
```

# ICON with Data Assimilation on GPU

- Kilometer-Scale Ensemble Data Assimilation (KENDA). Calculations in ensemble space spanned by the ensemble members, using observations to compute analysis
- Assimilation component takes the model and compares them with observations (DACE) – wirte feedback files (fof)



# Data Flow, and GPU strategy



- DACE code is not ported to GPU. The DACE code is kept on the CPU. Data is copied from GPU to CPU when needed.

xavier.lapillonne@meteoswiss.ch

# High level DSL for ICON

- Need to support unstructured grid, such as ICON grid
- New abstraction (e.g. neighbors operations)
- Focus on usability, productivity. Should be usable for domain scientist
- High level python dsl (gt4py)
- Development work in several projects, ESCAPE, EXCLAIM(ETHZ)

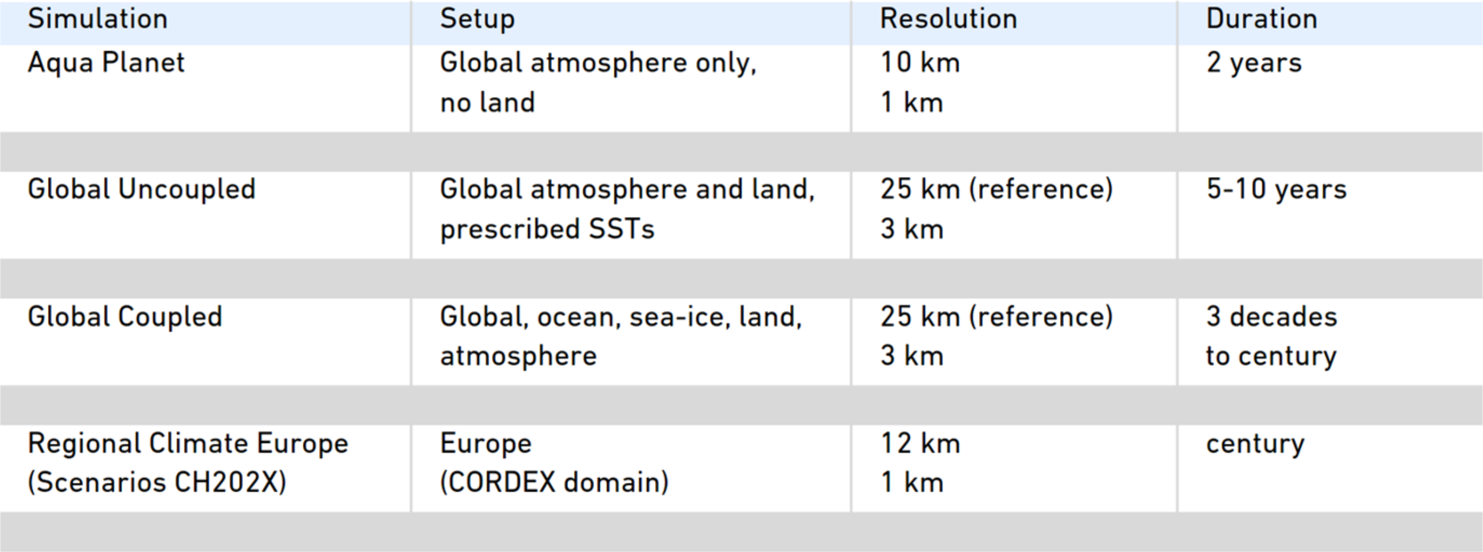
- Performance :
  - CUDA code generation for GPU : allow more optimization than OpenACC
  - long term perspective - data centric optimization across components, e.g. fusion …

xavier.lapillonne@meteoswiss.ch

# EXCLAIM Goals and Use Cases

Slides by Anurag Dipankar

Aims at developing an ICON- model based infrastructure, in particular using DSL, that is capable of running kilometer- scale climate simulations at both regional and global scales – C2SM, ETHZ, MeteoSwiss, CSCS



| Simulation | Setup | Resolution | Duration |
|---|---|---|---|
| Aqua Planet | Global atmosphere only, no land | 10 km<br>1 km | 2 years |
| Global Uncoupled | Global atmosphere and land, prescribed SSTs | 25 km (reference)<br>3 km | 5-10 years |
| Global Coupled | Global, ocean, sea-ice, land, atmosphere | 25 km (reference)<br>3 km | 3 decades to century |
| Regional Climate Europe (Scenarios CH202X) | Europe (CORDEX domain) | 12 km<br>1 km | century |

*Table 1: Overview of core scientific use cases*

# Motivation

$$\underline{\nabla_n}\psi(e) = \frac{\psi(c_1(e)) - \psi(c_0(e))}{\hat{l}}$$

```
#ifdef _OMP
!$OMP ....
#else
!$ACC ....
#endif
DO jb = i_startblk, i_endblk
 CALL get_indices_e(ptr_patch, ...)
 #ifdef __LOOP_EXCHANGE
 DO je = i_startidx, i_endidx
   DO jk = slev, elev
 #else
   DO jk = slev, elev
     DO je = i_startidx, i_endidx
 #endif
     grad_norm_psi_e(je,jk,jb) =  &
       ( psi_c(iidx(je,jb,2),jk,iblk(je,jb,2)) -
         psi_c(iidx(je,jb,1),jk,iblk(je,jb,1)) )
       / ptr_patch%edges%lhat(je,jb)
   ENDDO
 END DO
END DO
#ifdef _OMP
!$OMP ...
#else
!$ACC ...
#endif
```

# Motivation

$$\underline{\nabla_n}\psi(e) = \frac{\psi(c_1(e)) - \psi(c_0(e))}{\hat{l}}$$

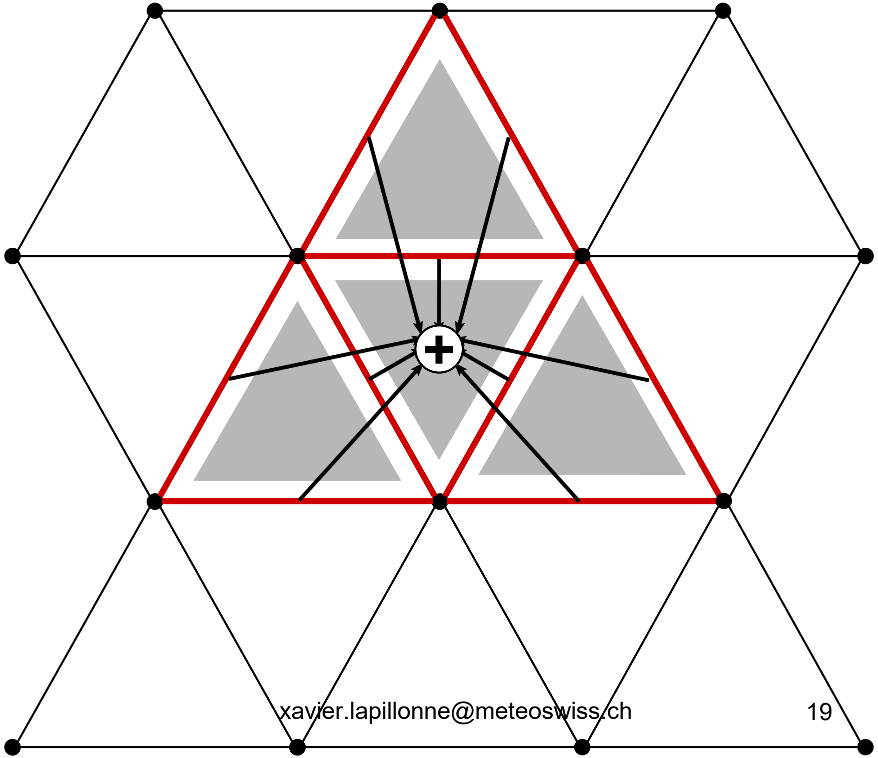
```
grad_norm_psi_e =
     sum_over(psi_c,
              Edge > Cell,
              [1/lhat, -1/lhat] )
```

```
#ifdef _OMP
!$OMP ....
#else
!$ACC ....
#endif
DO jb = i_startblk, i_endblk
 CALL get_indices_e(ptr_patch, ...)
 #ifdef __LOOP_EXCHANGE
 DO je = i_startidx, i_endidx
   DO jk = slev, elev
 #else
   DO jk = slev, elev
     DO je = i_startidx, i_endidx
 #endif
     grad_norm_psi_e(je,jk,jb) =  &
       ( psi_c(iidx(je,jb,2),jk,iblk(je,jb,2)) -
         psi_c(iidx(je,jb,1),jk,iblk(je,jb,1)) )
       / ptr_patch%edges%lhat(je,jb)
   ENDDO
 END DO
END DO
#ifdef _OMP
!$OMP ...
#else
!$ACC ...
#endif
```

xavier.lapillonne@meteoswiss.ch

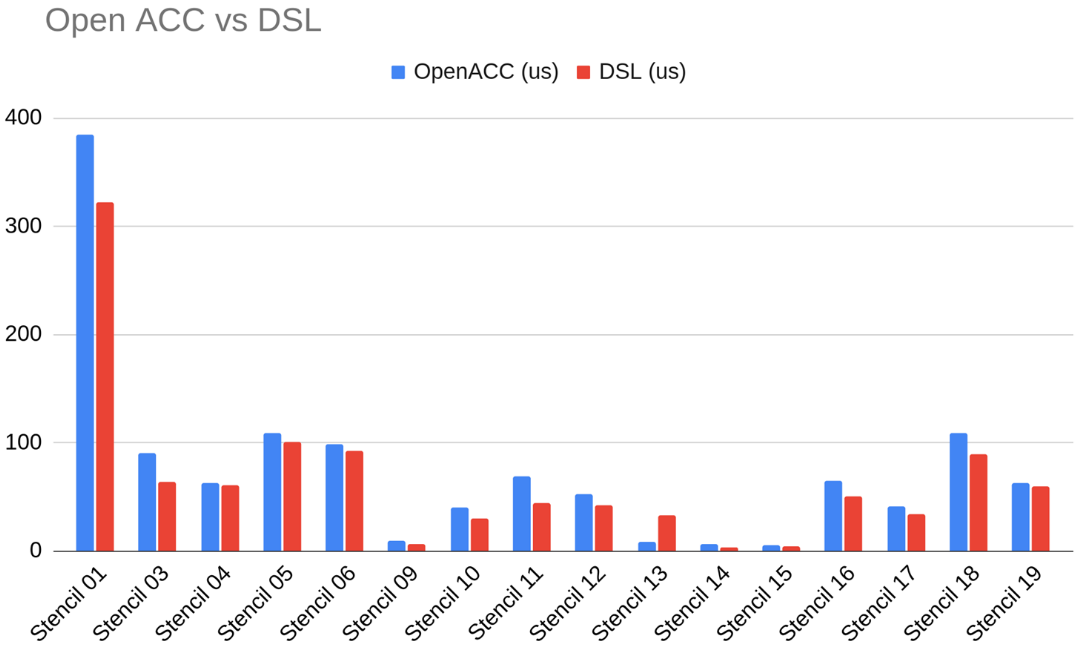
# Python DSL notation example (Dusk/Dawn) : Neighbor Chains

```
@stencil
def intp(fc: Field[Cell],
         fe: Field[Edge],
         w: Field[Cell > Edge > Cell > Edge]):
   with levels_downward:
       fc = sum_over(Cell > Edge > Cell > Edge,
                     w*fe)
```



xavier.lapillonne@meteoswiss.ch

# Performance of ICON dycore (DSL) prototype



DSL :
Dusk/
Dawn

- DSL dycore about 40% faster then OpenACC - not fully optimized. Dry dycore only.

# Conclusions

- COSMO model has been ported to GPU using DSL and OpenACC compiler directives, and is running in operation since 2016.
- Similar approach is considered for the ICON model, with a first version only based on compiler directives
- New High level DSL are being developed in particular in the EXCLAIM project
    - Separation of concern between scientist and computer engineer
    - More aggressive optimization and data centric optimization across components
    - Target more architectures
    - Improve usability and maintenance

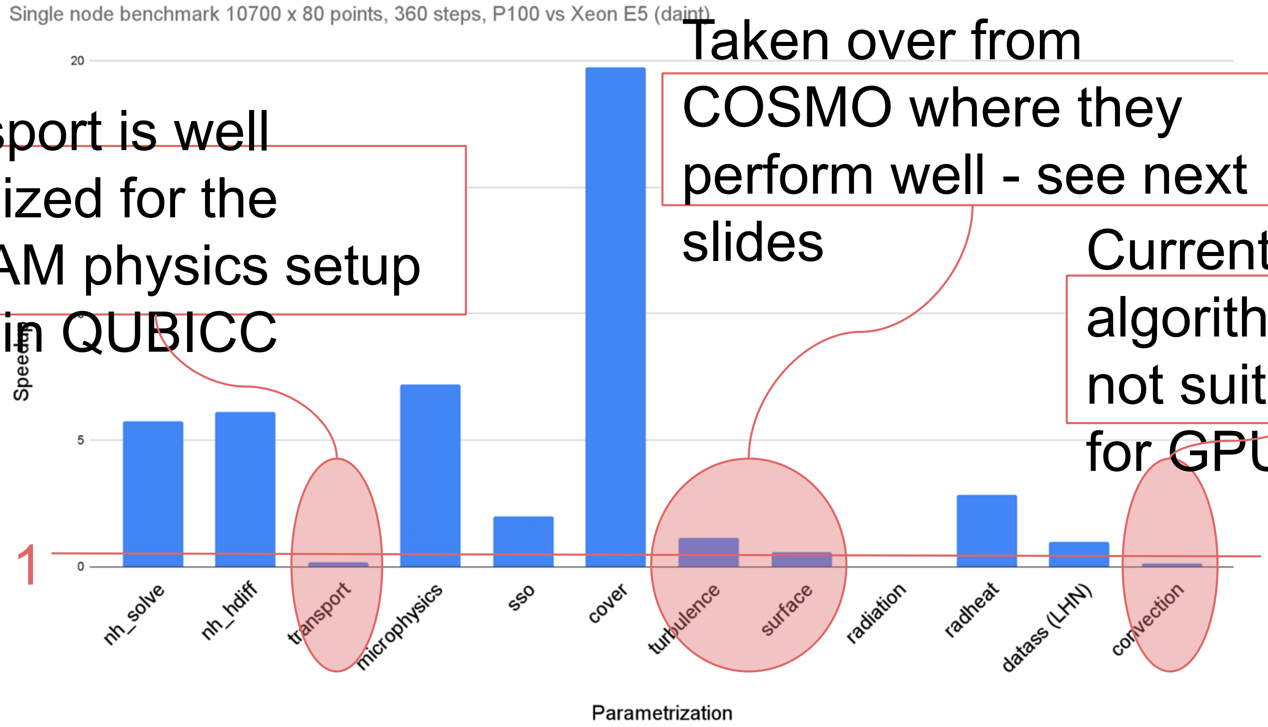
xavier.lapillonne@meteoswiss.ch

# Challenges using DSL in weather and climate

- How to bring along the scientific community: why changing ? Training, new learnings, keep productivity
- The quality level need to be high to keep user in the loop, need to be user friendly
- Need to achieve significant performance improvement to convince developers that this is the way forward
- Ensure and guarantee long term maintenance of the DSL infrastructure at a community level.
- DSL only work within the implemented pattern, how to let scientist have freedom to explore idea outside of this domain

# Additional slides

# Status of OpenACC performance



Transport is well optimized for the ECHAM physics setup used in QUBICC

Taken over from COSMO where they perform well - see next slides

Current algorithm likely not suited well for GPU

# C2SM / EXCLAIM project

EXascale Computing platform for cLoud-resolving weAther and clImate Models

**Goals:**

- develop an extreme scale computing and data platform for cloud resolving weather and climate modeling – prepare for exascale system
- redesign the codes in the framework of python base domain-specific languages
- exploit efficiency gains at both the computational and algorithmic levels
- develop an analysis and storage framework to deal with the exabytes of data generated
- design concrete applications addressing scale-interactions in the ocean and atmosphere
- Performance target : 1 simulated year / day @ 1 km global

Large project : 1. Senior scientist, 2 post-docs, 6 Software eng. + in in kind contributions from ETHZ, CSCS, EMPA, SDSC and MeteoSwiss

xavier.lapillonne@meteoswiss.ch

# Multi-core vs. GPU-accelerated hybrid

| | **Piz Dora (old code)** | **Piz Kesch (new code)** | Factor |
|---|---|---|---|
| Sockets | ~26 CPUs | ~7 GPUs | **3.7 x** |
| Energy | 10 kWh | 2.1 kWh | **4.8 x** |

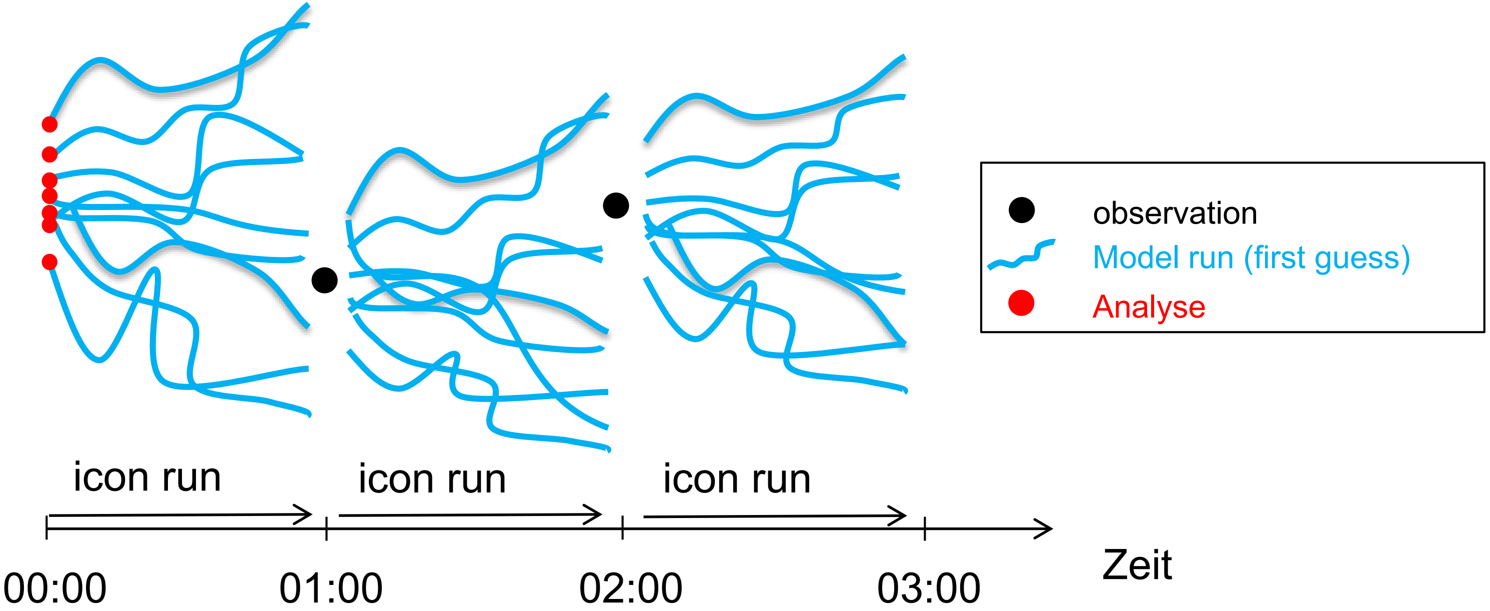
# Performance Results

- Test Setup as close as we can get right now to the operational setup:
  - Ca. 1.2M Grid columns (Area over Switzerland & Germany)
  - Simulation time of one hour
  - Run on Piz Daint 32 GPU nodes

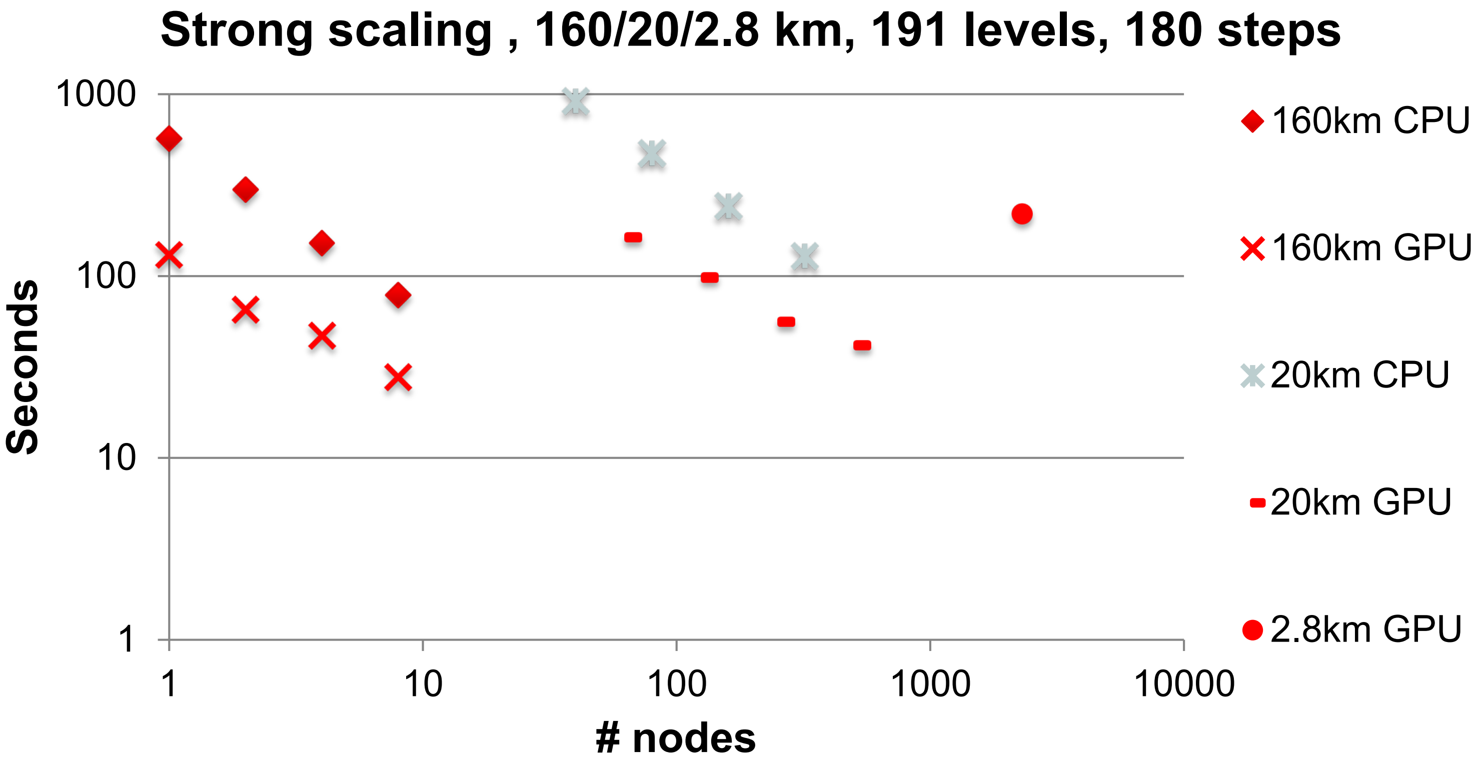
| | **GPU run** |
|---|---|
| ICON total [s] | **1214** |
| DACE total [s] | **6.95** |
| DACE total [%] | **0.57** |

xavier.lapillonne@meteoswiss.ch

# KENDA Assimilationszyklus



# Performance : strong scaling (QUBICC conf.)



CPU: 1xHaswell, 12 ranks/node, OMP=2 (CCE)
GPU: 1xP100 (daint-gpu), 2 ranks/node (PGI)

xavier.lapillonne@meteoswiss.ch