7. HPC-Status-Konferenz der Gauß-Allianz in Stuttgart

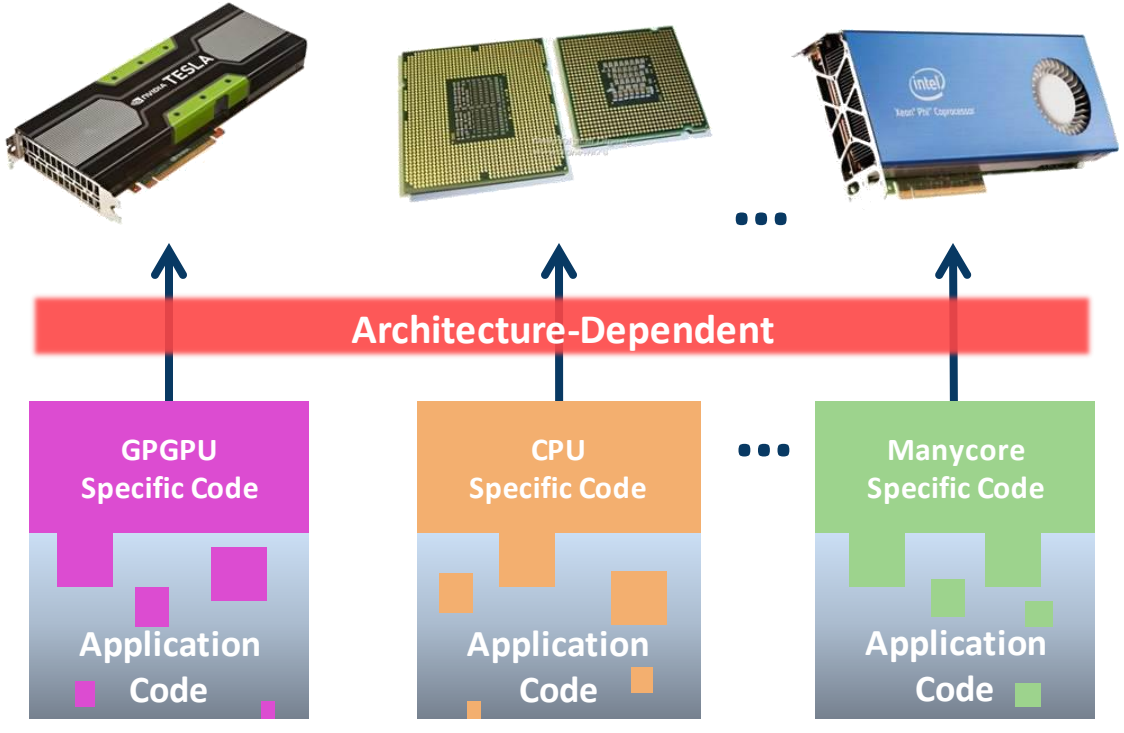
MEPHISTO: FUTURE PROOF HPC APPLICATION PROGRAMMING

Dr. Andreas Knüpfer & Bert Wesarg, TU Dresden

Outline

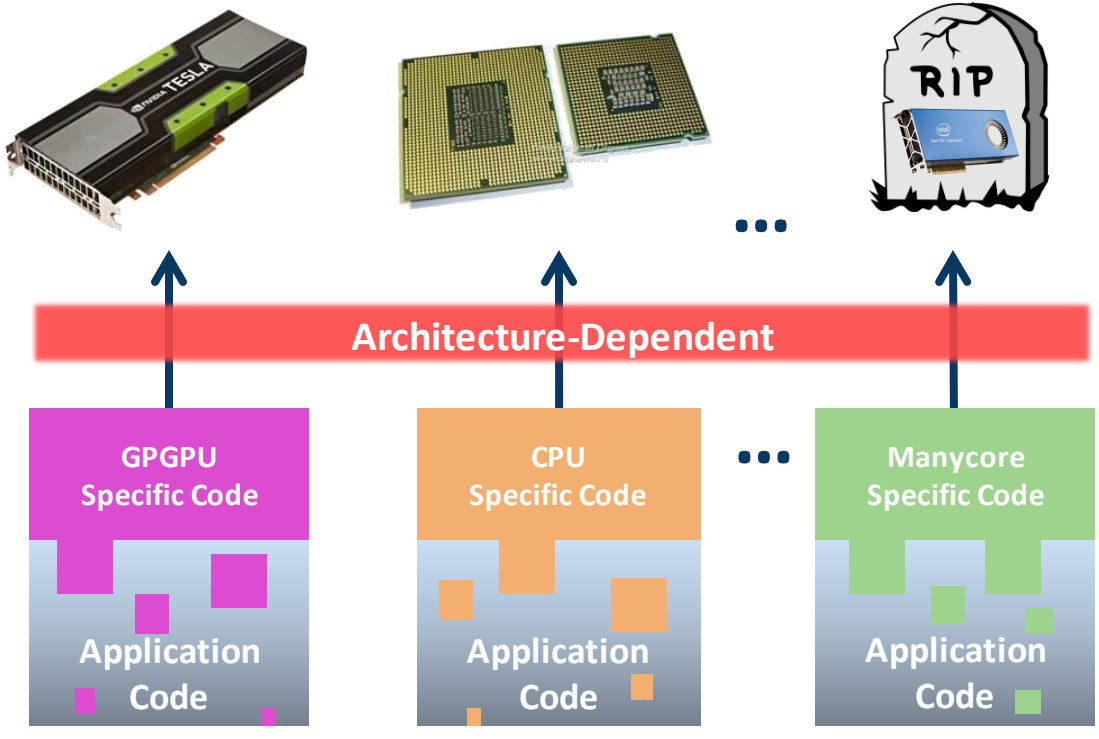
- Challenges in writing HPC applications
- The MEPHISTO project
- Fellow contenders

Complexity of HPC systems



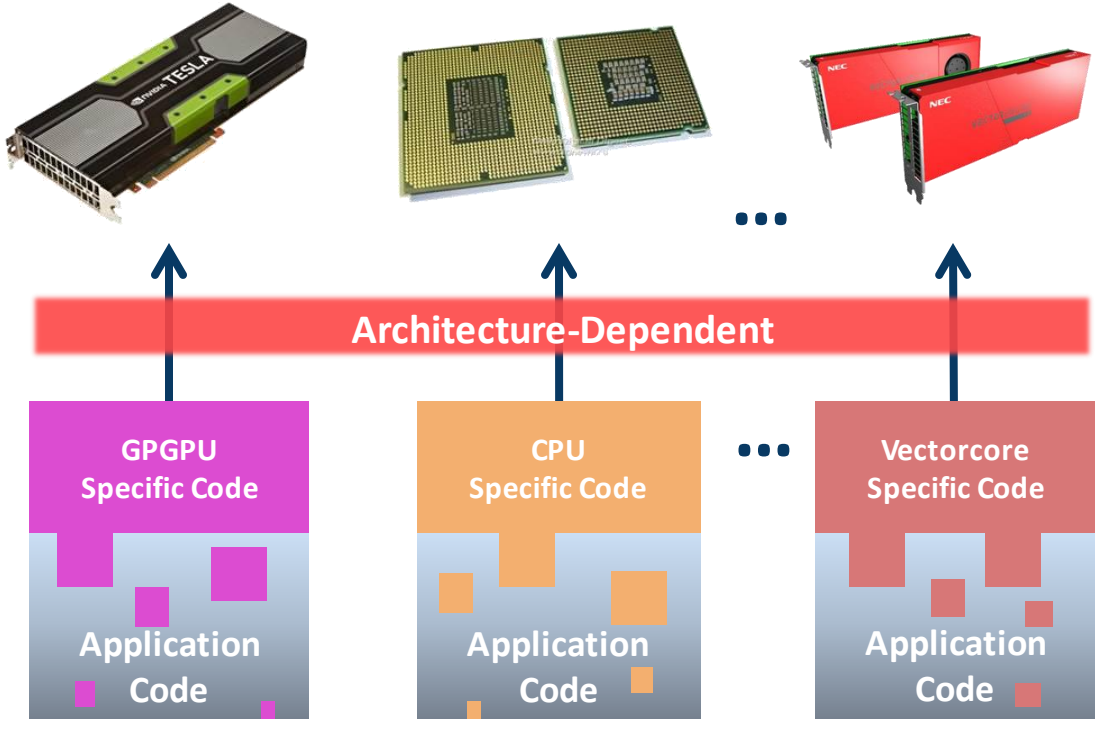
- CPUs and Accelerators
  - CUDA, Xeon Phi, NEC SX-Aurora
- CPU ISA
- Storage hierarchy
  - L1/L2/L3 caches
  - RAM, NUMA
  - NV Memory
  - Disk/SSD/Tape/…
- Interconnect

Complexity of HPC systems



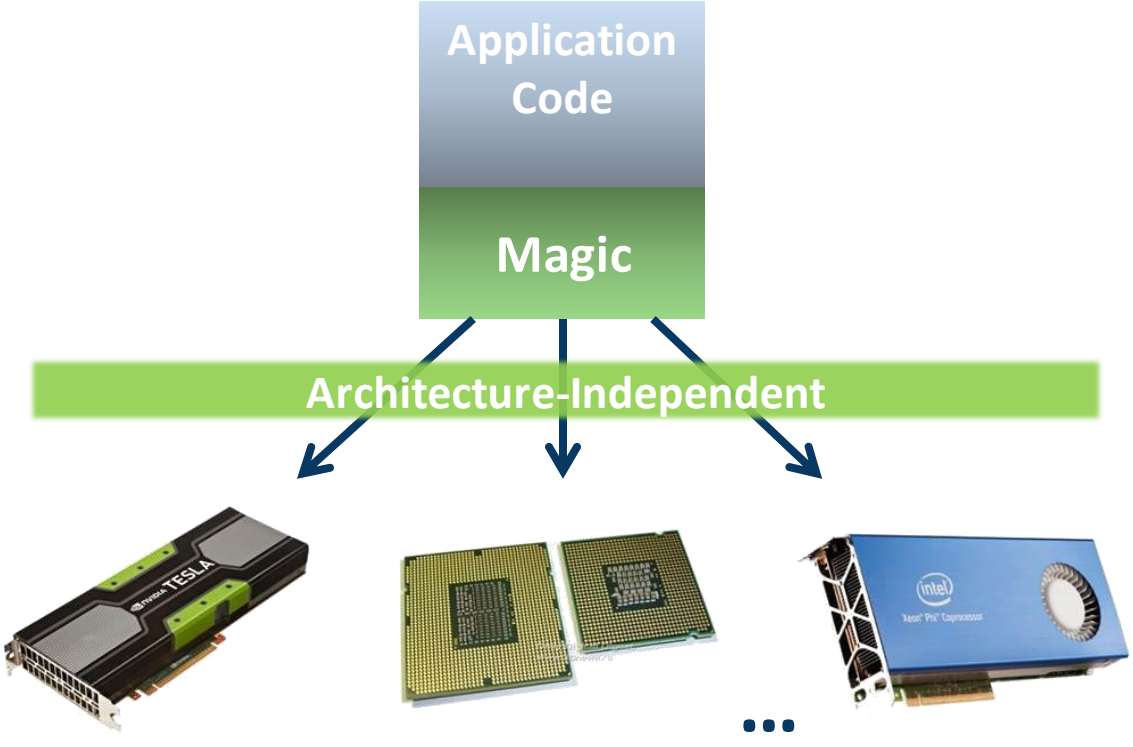
- CPUs and Accelerators
  - CUDA, Xeon Phi, NEC SX-Aurora
- CPU ISA
- Storage hierarchy
  - L1/L2/L3 caches
  - RAM, NUMA
  - NV Memory
  - Disk/SSD/Tape/…
- Interconnect

Complexity of HPC systems



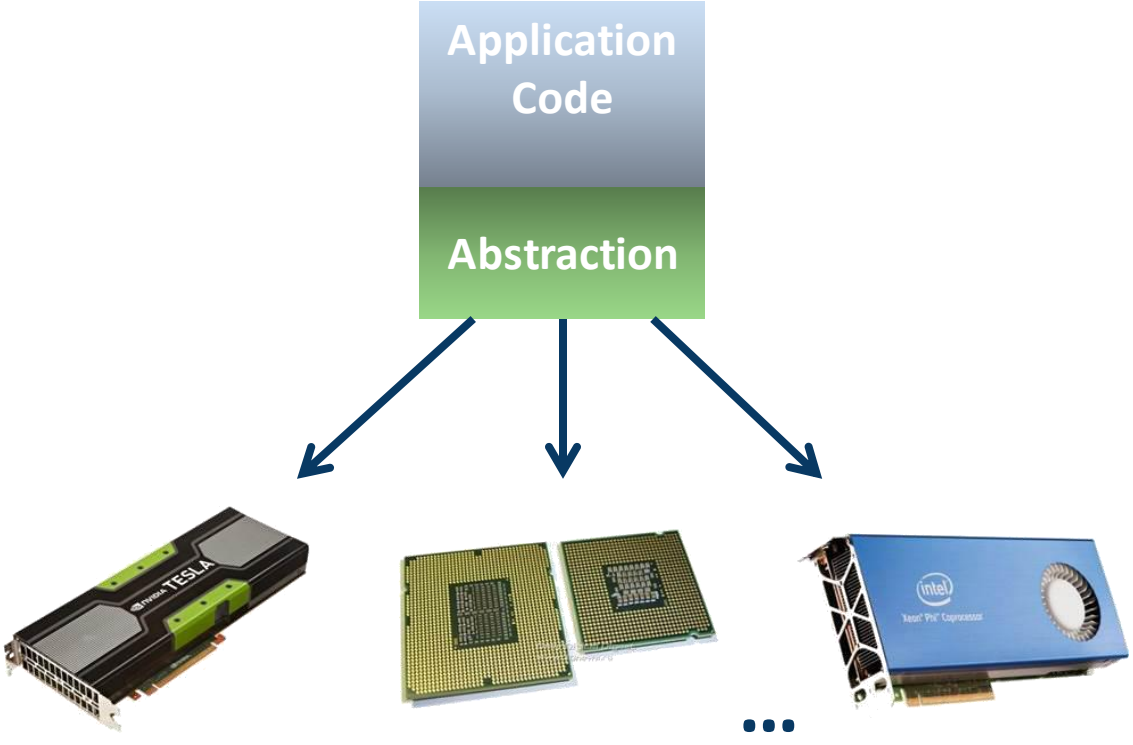
- CPUs and Accelerators
  - CUDA, Xeon Phi, NEC SX-Aurora
- CPU ISA
- Storage hierarchy
  - L1/L2/L3 caches
  - RAM, NUMA
  - NV Memory
  - Disk/SSD/Tape/…
- Interconnect

The ideal HPC application



- Maintainable
- Portable
- Scalable

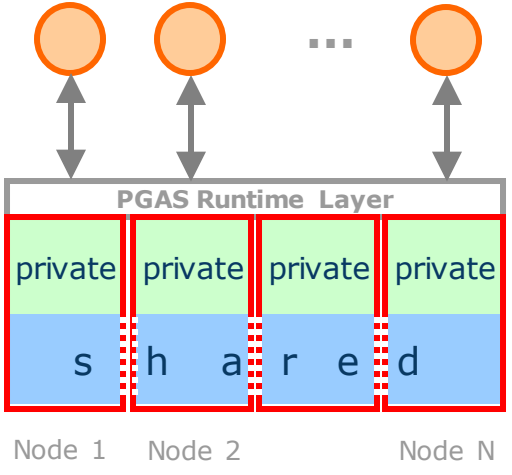
The ideal HPC application



- Maintainable
- Portable
- Scalable

Inter-node complexity

- Partition Global Address Space (PGAS)
- Reduce communication
- Consider heterogeneous architectures
- No “MPI+X” in application code



Mephisto: Its all about abstractions

- MPI abstracts network
- MPI programming still hard and error prone

→ Use MPI abstraction libraries

- Plenty of node-level programming models

→ Best fit depends on multiple factors

→ Use abstraction to select best programming model

- Native C++ 11/14 “look and feel”

MEPHISTO

- C++ meta programming for heterogeneous distributed systems
- Do not reinvent the wheel, combine existing solutions to provide wholesale abstraction

SPONSORED BY THE
Federal Ministry of Education and Research

Partners

- Technische Universität Dresden (TUD)
- Ludwig-Maximilians-Universität München (LMU)
- Helmholtz-Zentrum Dresden Rossendorf (HZDR)

TECHNISCHE UNIVERSITÄT DRESDEN

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

HZDR
HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF

Existing solution: DASH for inter-node abstraction

- C++ Template Library for Distributed Data Structures
- PGAS (Partitioned Global Address Space) libraries for one-sided global access

Gefördert durch
DFG Deutsche Forschungsgemeinschaft

Partners

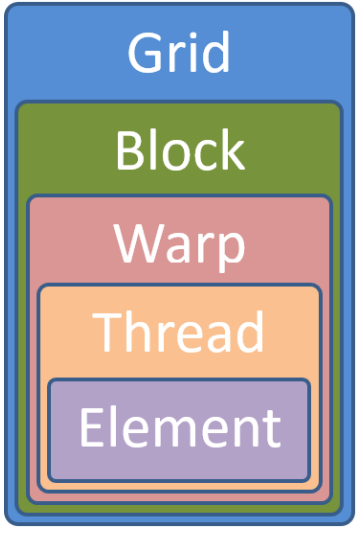
- Ludwig-Maximilians-Universität München (LMU)
- Technische Universität Dresden (TUD)
- Höchstleistungsrechenzentrum Stuttgart (HLRS)

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

TECHNISCHE UNIVERSITÄT DRESDEN

HLRS

Existing solution: Alpaka for node-level abstraction



- C++ node-level programming abstraction library
- Developed by Computational Radiation Physics group at Helmholtz-Zentrum Dresden Rossendorf (HZDR)
- Supported backends
  - Serial CPU
  - OpenMP, Intel Threading Blocks
  - CUDA, OpenACC

HZDR

HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF

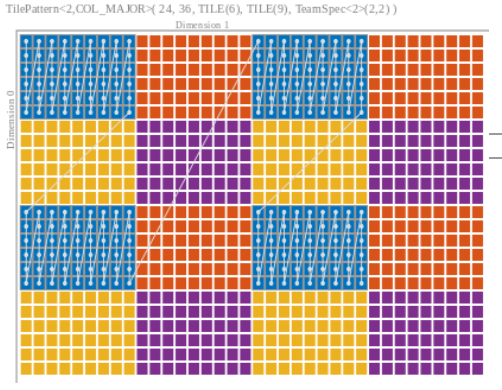
```
NArray<int,2> matrix( {24,36}, DistributionSpec<2>( ... ) );

for ( auto el: matrix ) {
    cout << (int) el << " ";
}

for ( auto it= matrix.begin(); it != matrix.end(); ++it ) {
    sum += *it;
}

for ( auto it= matrix.lbegin(); it != matrix.lend(); ++it ) {
    sum += *it;
}

std::fill( matrix.lbegin(), matrix.lend(), 0 );
dash::fill( matrix.begin(), matrix.end(), 1 );
```



MEPHISTO: Goal and project status

- Avoid "DASH+Alpaka" along the lines of "MPI+X"
- Strive for synthesis of DASH and Alpaka

  → Leveraging C++ idioms on all levels of parallelism
- Demonstrating applicability on:
  - Port of LULESH benchmark
  - Multi-grid solver of heat dispersion equation (video)

- DASH/Alpaka interface defined
- DASH distribution pattern specification for node-level memory layout
- Considering SoA vs. AoS extension of DASH distribution patterns

Related work

Kokkos – C++ performance portable programming

- Developed by Sandia National Laboratories since 2014
- Results in similar performance as Alpaka
- Provides its own array abstraction comparable to DASH patterns

HPX - C++ runtime system for parallel and distributed applications

- Developed by the STE||AR Group (LSU, SNL, NMSU, LBL)
- Own runtime system (AGAS), not based on MPI
- Impressive performance and scalability demonstations
- Misses "sensible defaults with least astonishment"

MEPHISTO

- Abstraction of distributed data structures (inter-node)
- Abstraction of node-level parallel execution (intra-node)
- … both with C++ template meta programming at compile time

- Modern and native C++
- Wholesale parallel programming experience
- Less complexity for application codes