A Case Study on Multi-Component Multi-Cluster Interaction with an AMR Solver

WOLFHPC 2013

A. Atanasov, H.-J. Bungartz, K. Unterweger,T. Weinzierl, R. Wittmann

November, 18th 2013

Outline

Motivation

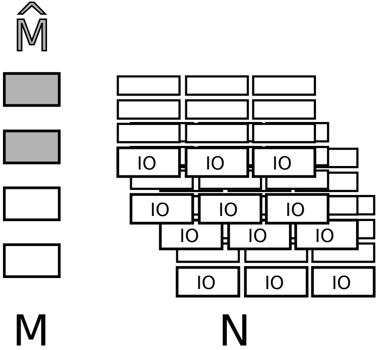
Communication

Software Architecture

Results

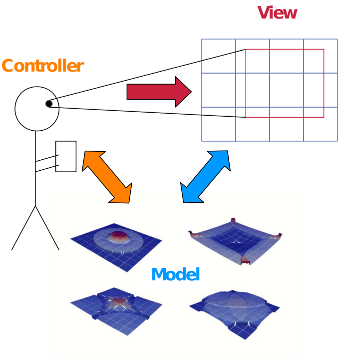
Conclusion & Outlook

Motivation



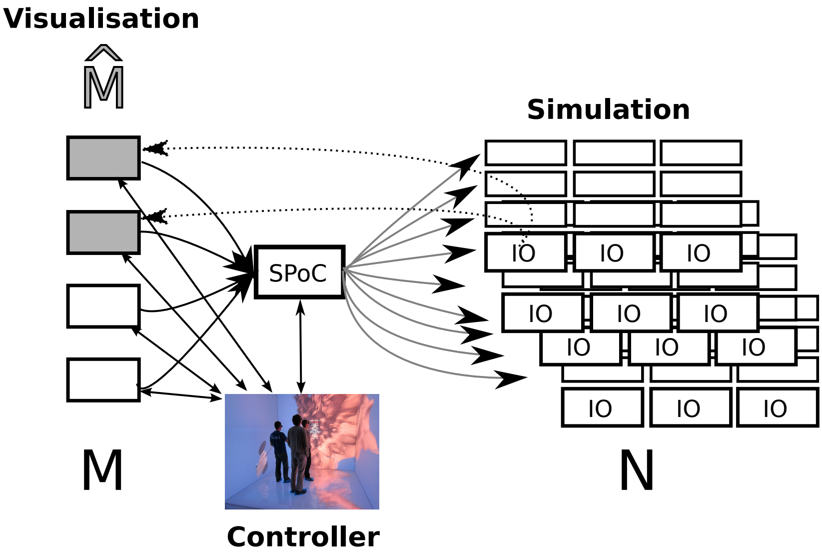
- Multi-cluster interactions between non-monolithic applications
- Examples : Multi-Physics, simulation-postprocessing coupling, ...
- M request sources, N data sources, $\hat{M}$ destination nodes

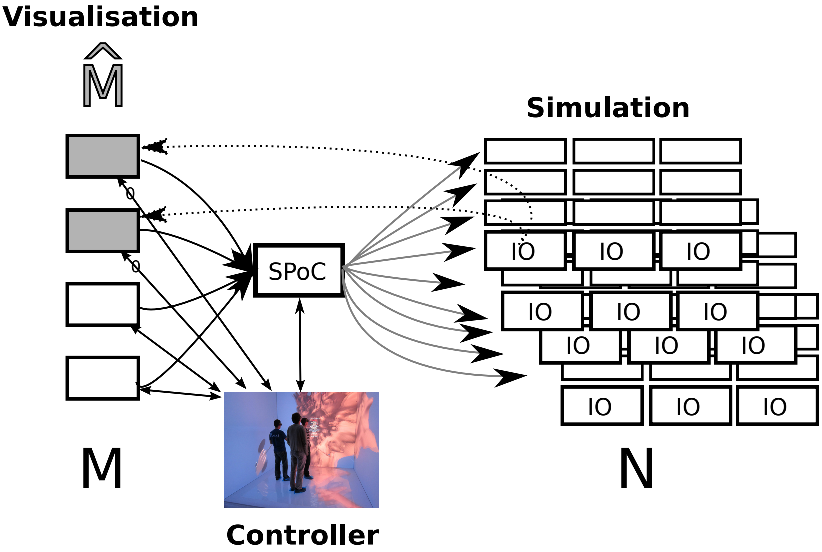
Use Case

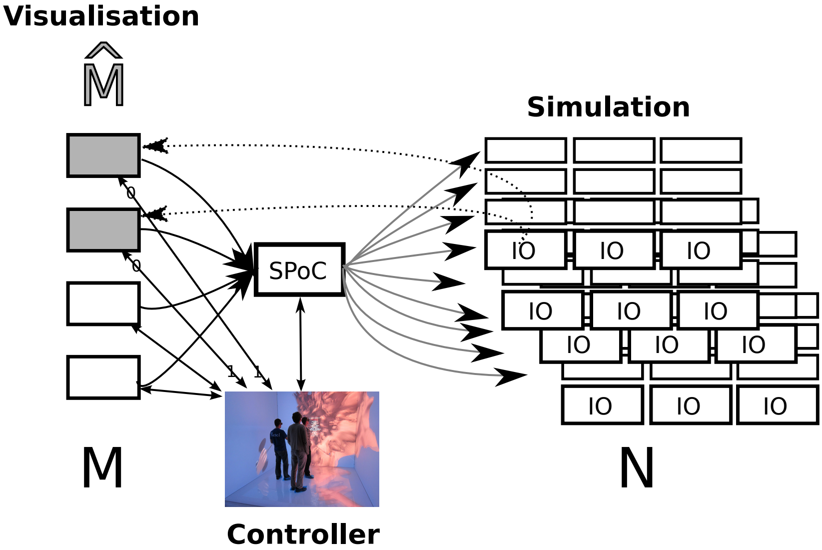


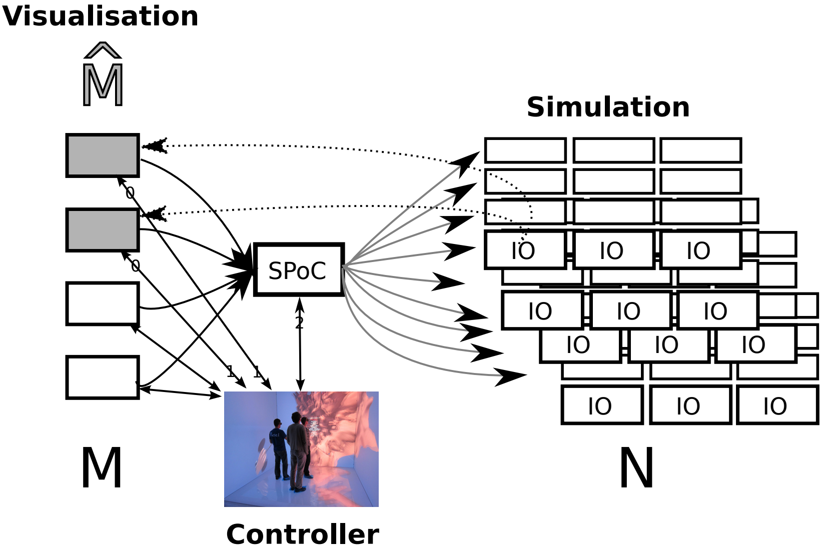
- Simulation Model: Parallel AMR solver for hyperbolic PDEs [4]
- Visualisation: Query-based data retrieval [3]
- Controller : User located in VR environment

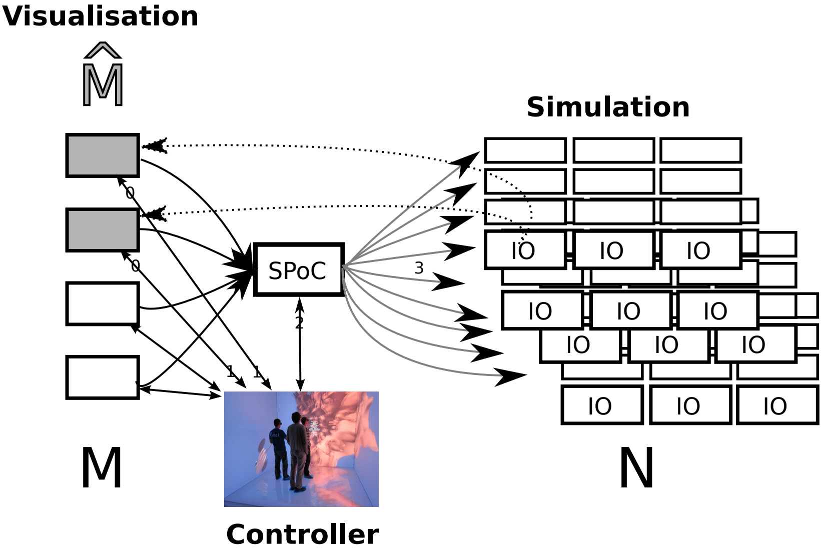
Outline


Visualisation
M̂
Simulation
SPoC
IO IO IO
IO IO IO
IO IO IO
IO IO IO
M
N
Controller

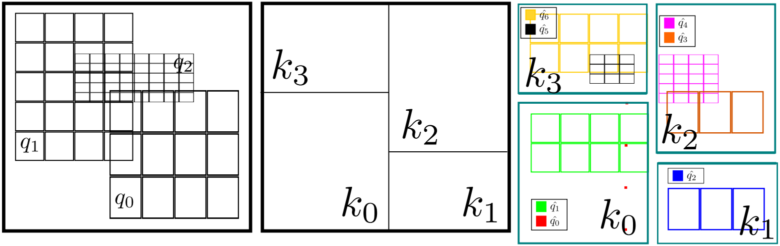

Visualisation
$\hat{M}$
Simulation
0
0
SPoC
IO IO IO
IO IO IO
IO IO IO
IO IO IO
M
N
Controller


Visualisation
M̂
Simulation
0
0
SPoC
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
1
1
M
N
Controller

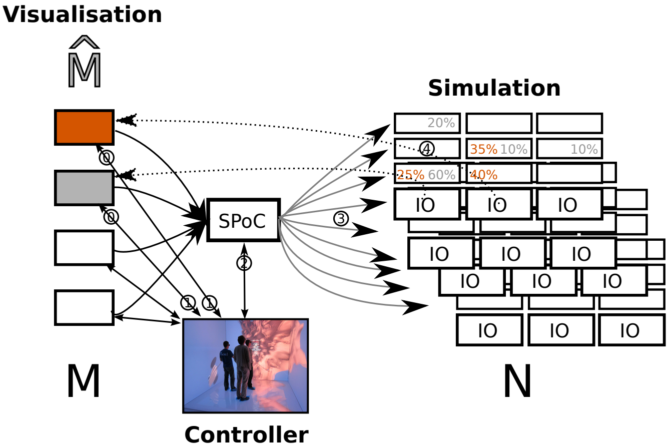

Visualisation
$\hat{M}$
Simulation
SPoC
0
0
1
1
2
IO
M
N
Controller


Visualisation
$\hat{M}$
Simulation
SPoC
0
0
1
1
2
3
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
M
N
Controller

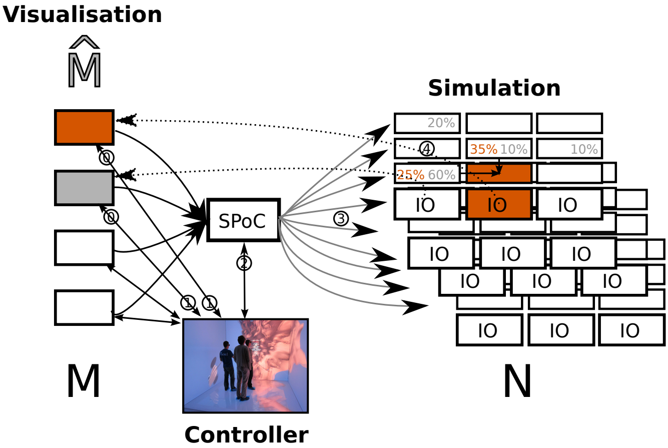
Routing Information $(0-3)$



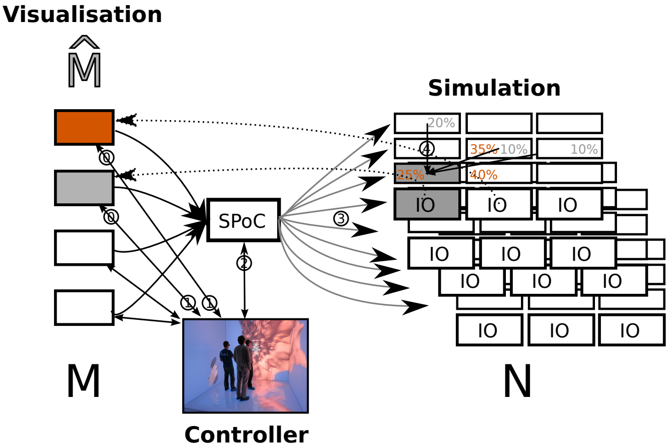
Merges : Greedy Merge *Version*$(4+5)$



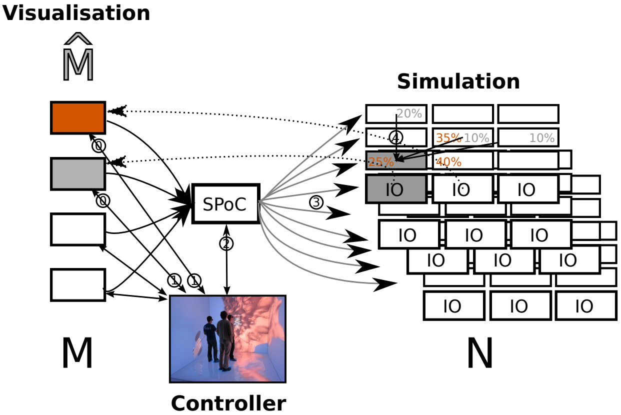
Merges : Greedy Merge $Version(4+5)$



Merges : Greedy Merge *Version*(4 + 5)

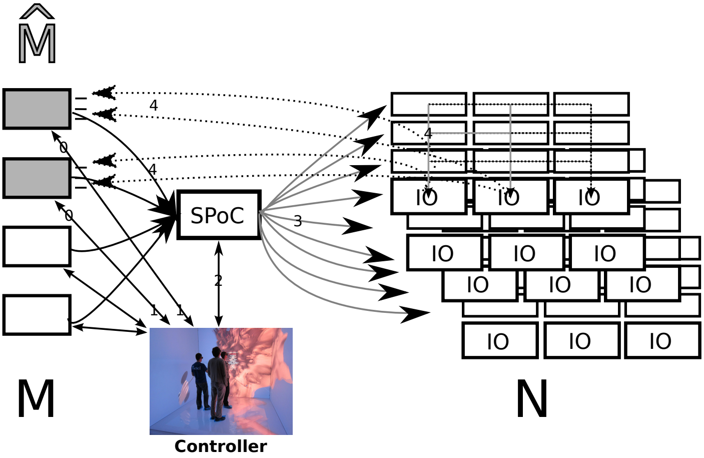


Merges : Greedy Merge *Version*$(4+5)$

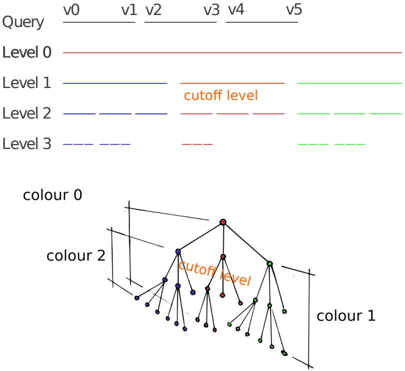


- Global identification of the nodes holding most data
- Merge data on identified nodes
- Forward merged data to destinations

Visualisation Simulation



Hierarchical Merge *Version*(4)



- Use a logical tree to merge data
- Merge data up to given cutoff ℓ
- Destinations merge the remaining data fragments

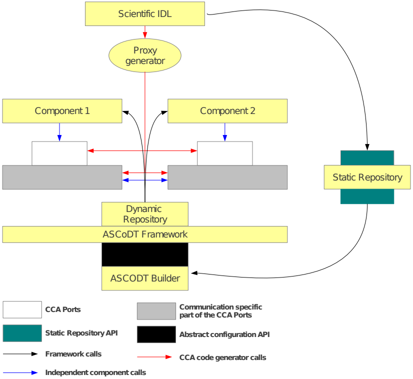
Tree Merge, cont.

Properties

- Requires a logical master-worker hierarchy
- Fully asynchronous
- Additional memory requirements dependent on ℓ
- ℓ can be chosen dynamically
- Merge distributed between N and $\hat{M}$

Outline

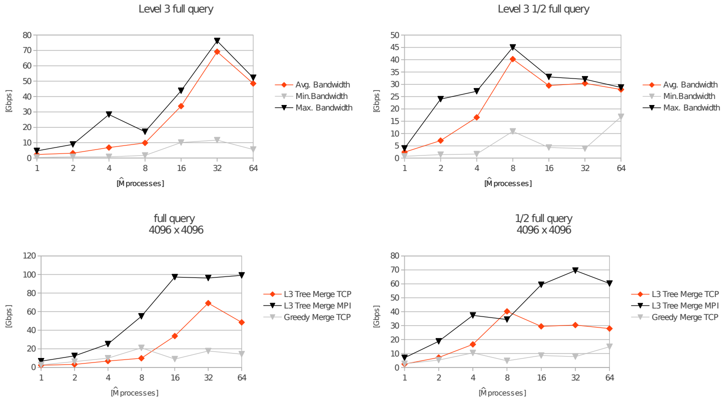
Software Architecture



- A lightweighted distributed framework based on CCA [1]
- Communication through BSD sockets or MPI Ports
- IDE for component development [2]
- Generate glue code from SIDL annotations

Outline

Results



- Fully refined computational domain with 9^8 (6561×6561) cells
- Two queries : full domain query, half domain query (4096×4096)

Outline

Conclusion & Outlook

Enabling technologies

- On-demand data retrieval
- Destination-aware data routing
- Asynchronous partial merges
- Hide technical details through CCA

Outlook

- Autotuning approaches
- User-defined communication protocols
- Multi-physics applications

References I

R. Armstrong, D. Gannon, A. Geist, K. Keahey, S. Kohn, L. McInnes, S. Parker, and B. Smolinski.
Toward a common component architecture for high-performance scientific computing.
In *High Performance Distributed Computing, 1999. Proceedings. The Eighth International Symposium on*, pages 115–124. IEEE, 1999.

A. Atanasov, H.-J. Bungartz, and T. Weinzierl.
A toolkit for the code development in advanced computing.
Technical Report TUM-I1330, 2013.

References II

- A. Atanasov, M. Srinivasan, and T. Weinzierl.
  Query-driven parallel exploration of large datasets.
  In *Large Data Analysis and Visualization (LDAV), 2012 IEEE Symposium on*, pages 23 –30, Oct. 2012.
  doi 10.1109/LDAV.2012.6378972.

- K. Unterweger, T. Weinzierl, D. I. Ketcheson, and A. Ahmadia.
  Peanoclaw—a functionally-decomposed approach to adaptive mesh refinement with local time stepping for hyperbolic conservation law solvers.
  Technische Universität München, Technical Reports, 2013.