High-Performance Distributed Multi-Model / Multi-Kernel Simulations: 
A Case Study in Jungle Computing Using Ibis 

Frank J. Seinstra 

Jungle Computing Research & Applications Group 
Department of Computer Science 
VU University, Amsterdam, The Netherlands
The Ibis e-Science Software Framework

Frank J. Seinstra

Jungle Computing Research & Applications Group
Department of Computer Science
VU University, Amsterdam, The Netherlands
eScience - Recent Developments
(in The Netherlands)
Amsterdam

- A center of HPC, networking, eScience (zoom level 1)

Source: GLIF – Global Lambda Integrated Facility

COST Workshop 2012 - Timisoara
Amsterdam Science Park

- A center of HPC, networking, eScience (zoom level 2)
Netherlands eScience Center

- A center of HPC, networking, eScience (zoom level 3)

- Science & Science Support (ICT)
- Priority Areas:
  - Astronomy
  - Climate Research
  - Cognition
  - Chemistry
  - Ecology
  - eScience Technologies
  - Humanities
  - Food Research
  - Green Genetics
  - Life Sciences
  - Water Management

- Selected Key Technology:
  - Ibis / Jungle Computing
Ibis: ‘Problem Solving’ vs. ‘System Fighting’
A Random Example: **Supernova Detection**

- **DACH 2008, Japan**
  - Distributed multi-cluster system
    - Heterogeneous
  - Distributed database (image pairs)
    - Large vs small databases/images
    - Partial replication
  - Image-pair comparison given (in C)

- **Find all supernova candidates**
  - Task 1: As fast as possible
  - Task 2: Idem, under system crashes
‘Problem Solving’ vs. ‘System Fighting’

- All participating teams struggled (1 month)
  - Middleware instabilities…
  - Connectivity problems…
  - Load balancing…

- But not the Ibis team
  - Winner (by far) in both categories
  - Note: many Japanese teams with years of experience
    - Hardware, middleware, network, C-code, image data…
  - Focus on ‘problem solving’, not ‘system fighting’
    - incl. ‘opening’ of black-box C-code
Ibis Results: Awards & Prizes

- AAAI-VC 2007
  Most Visionary Research Award

1st Prize: SCALE 2008 – BS
(scalability)

1st Prize: SCALE 2010
(scalability)

1st Prize: DACH 2008 – FT
(fault-tolerance)

1st Prize: EYR3 2011
(sustainability)

- Many domains; data/compute intensive, real-time...
Ibis Users...

...and many more
Jungle Computing
Jungle Computing

- ‘Worst case’ computing as required by end-users
  - Distributed
  - Heterogeneous
  - Hierarchical (incl. multi-/many-cores)
Why Jungle Computing?

- Scientists often *forced* to use a wide variety of resources *simultaneously* to solve computational problems, e.g. due to:
  - Desire for scalability
  - Distributed nature of (input) data
  - Software heterogeneity (e.g.: mix of C/MPI and CUDA)
  - Ad hoc hardware availability
  - …

- Note: most users do not need ‘worst case’ jungle
  - Ibis aims to apply to any subset
Problems in the Jungle

- Jungle Computing for domain scientists?
  - Hardware heterogeneity
  - Middleware heterogeneity
  - Software heterogeneity
    - Kernels in C, MPI, Fortran, Java, CUDA, scripts, …
  - Connectivity problems
    - e.g. firewalls, NAT, …
  - Infrastructure often dynamic, faulty
  - ….

- Need for integrated, user-friendly solution/toolbox
  - Focus on ‘problem solving’, not ‘system fighting’
The Ibis Software Framework

Applications

Ibis High-Performance Programming System
- Programming Models
- Ibis Portability Layer (IPL)

Ibis Distributed Deployment System
- IbisDeploy (GUI)
- JavaGAT
- Zorilla

Low-Level Communication Protocols & Computing Hardware

Jungle-aware communication
transparently overcome connection setup problems
middleware independence
Jungle-aware middleware
Domain Example #1:
Computational Astrophysics

with: Prof. Simon Portegies Zwart and Inti Pelupessy
(Leiden Observatory / Leiden University)
Domain Example #1: **Computational Astrophysics**

See: [http://www.youtube.com/watch?v=yE8LL1rE880](http://www.youtube.com/watch?v=yE8LL1rE880)

Demonstrated live at SC’11, Nov 12-18, 2011, Seattle, USA  (two months ago)
Domain Example #1: **Computational Astrophysics**

- **The AMUSE system (Leiden University)**
  - Early Star Cluster Evolution, including gas

- **AMUSE**
  - **Gravitational dynamics**
  - **Stellar evolution**
  - **Radiative transport**
  - **Hydro-dynamics**

- **Gravitational dynamics (N-body):** GPU / GPU-cluster
- **Stellar evolution:** Beowulf cluster / Cloud
- **Hydro-dynamics, Radiative transport:** Supercomputer
Domain Example #1: Computational Astrophysics

Demonstrated live at SC'11, Nov 12-18, 2011, Seattle, USA
Domain Example #1: **Computational Astrophysics**

See: [http://www.youtube.com/watch?v=_mm0lq8LJMcg](http://www.youtube.com/watch?v=_mm0lq8LJMcg)

Demonstrated live at SC’11, Nov 12-18, 2011, Seattle, USA (two months ago)
Domain Example #2: Climate Modeling

with: Prof. Henk Dijkstra and Michael Kliphuis (Utrecht University)
Domain Example #2: Climate Modeling

- The CPL system (Utrecht University)
  - or: The Community Earth System Model (CESM)

- Ocean, Sea-ice
- Atmosphere, Land-vegetation

Graphical representation:

- CPL
- atmosphere
- ocean
- sea-ice
- land-vegetation

GPU / GPU-cluster
cluster / Cloud, or supercomputer
Three Common Uses of Ibis
Ibis as ‘Master Key’ (or ‘Passepartout’)

- Use JavaGAT to access ‘any’ system
  - Develop/run applications independently of available middlewares
  - JavaGAT ‘adaptors’ required for each middleware
  - ‘Intelligent dispatching’ even allows for transparent use of multiple middlewares

- Example: file copy
  - JavaGAT vs. Globus
    - Simple, portable, …
    - SAGA API standardized
Ibis as ‘Glue’

- Use IPL + SmartSockets, generally for wide-area communication
  - Linking up separate ‘activities’ of an application
    - Activities: often largely ‘independent’ tasks implemented in any popular language or model (e.g. C/MPI, CUDA, Fortran, Java…)
    - Each typically running on a single GPU/node/Cluster/Cloud/…
  - Automatically circumvent connectivity problems
- Example:

With SmartSockets: ![Network Diagram]

No SmartSockets: ![Network Diagram]
Ibis as ‘HPC Solution’

- Use Ibis as replacement for e.g. C++/MPI code
  - Benefits:
    - (better) portability
    - malleability (open world)
    - fault-tolerance
    - (run-time) task migration
  - Downside:
    - requires recoding
- Comparable speedups:
Domain Example #3: Multimedia Content Analysis

with: Prof. Arnold Smeulders and Dr. Jan-Mark Geusebroek
(University of Amsterdam)
Domain Example #3: Color-based Object Recognition by a Grid-connected Robot Dog

Seinstra et al (AAAI’07: Most Visionary Research Award)
See: http://www.cs.vu.nl/~fjseins/AiboDemo/AiboRecognize.shtml
‘Master Key’ + ‘Glue’ + ‘HPC’

  ● ‘Multimedia services’ in C++/MPI (pre-installed at each cluster)
  ● Wide-area communication using TCP Sockets (instable)
  ● SSH tunneling where necessary (firewalls, …)
  ● Execution on each cluster ‘by hand’

● Step-wise conversion to 100% Ibis / Java
  ● Phase 1: JavaGAT as ‘Master Key’
  ● Phase 2: IPL + SmartSockets as ‘Glue’
  ● Phase 3: Ibis as ‘HPC Solution’
  ● Eventual result:
    ● ‘wall-socket computing from a memory stick’
    ● Awards at AAAI 2007 and CCGrid 2008
Conclusions

- Ibis enables problem solving (avoids system fighting)
- Successfully applied in many domains
  - Astronomy, multimedia analysis, climate modeling, remote sensing, semantic web, medical imaging, ...
  - Data intensive, compute intensive, real-time...
- One of key technologies selected by NLeSC
- Open source, download:
  - www.cs.vu.nl/ibis/
Thank You!

www.cs.vu.nl/ibis/