The Ibis e-Science Software Framework

Frank J. Seinstra, Jason Maassen, Niels Drost

Jungle Computing Research & Applications Group
Department of Computer Science
VU University, Amsterdam, The Netherlands

vrije Universiteit amsterdam

Mini-course: General Overview

- 10:00 – 11:00: Introduction to Ibis
- 11:00 – 12:00: Ibis as ‘Master Key’
- 12:00 – 13:00: Lunch (free)
- 13:00 – 14:00: Ibis as ‘Glue’
- 14:00 – 15:00: Ibis today, tomorrow, and beyond
- During and after course time for discussion

- Note:
  - Second day (hands-on session) on Monday December 12

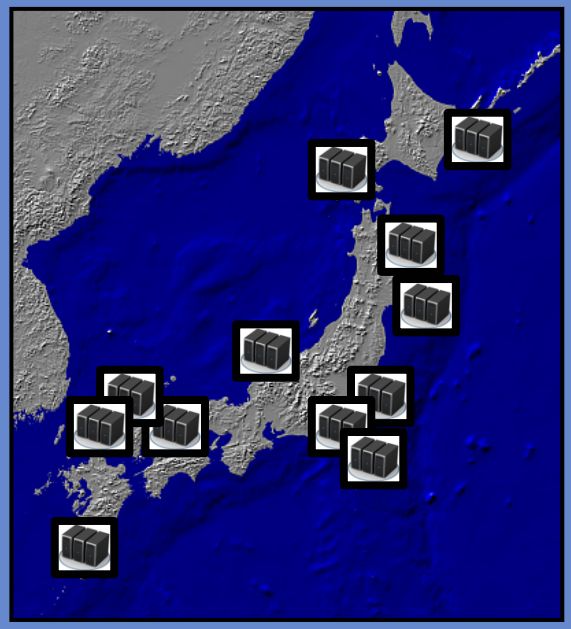
Introduction: Overview

- Ibis: ‘Problem Solving’ vs. ‘System Fighting’
- Disclaimers
- Jungle Computing
- Domain Example #1: Computational Astrophysics
- Domain Example #2: Multimedia Content Analysis
- The Ibis Software Framework
  - Requirements and Tools
- The 3 Common Uses of Ibis
  - Ibis as ‘Master Key’, Ibis as ‘Glue’, Ibis as ‘HPC solution’

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’

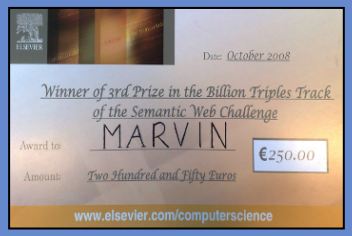
- All participating teams struggled (1 month)
  - Middleware instabilities…
  - Connectivity problems…
  - Load balancing…
- But not the Ibis team
  - Winner (by far) in *both* categories
  - Note: many Japanese teams with years of experience
    - Hardware, middleware, network, C-code, image data…
  - 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: DACH 2008 - BS

1st Prize: DACH 2008 - FT



3rd Prize: ISWC 2008

1st Prize: SCALE 2008

1st Prize: SCALE 2010

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

Finalist: EYR3 2011

Ibis Users…

...and many more

Disclaimers

Disclaimer #1

- During this mini-course I (= Frank) will take the position of an enthusiastic user / domain expert…
  - Because I am / used to be:
    - Multimedia Content Analysis (UvA, 1996 - 2006)
  - 2005/2006:
    - Ibis provided all tools to allow me to continue my research
    - Was immediately successful: award at AAAI 2007
    - Became Ibis ‘evangelist’: “If I can do it, anyone can”
  - Now leader of JCRA group
    - Direction, focus, scope
    - Outreach to application domains / developers

Disclaimer #2

- The Ibis projected started in 2002
  - Dozens of people have worked on it and/or co-implemented it
  - Many more have used it
  - Hence:
    - Ibis reflects many important lessons learned over +/- 10 years
- In this tutorial we can only ‘scratch the surface’
  - Attempt to give best possible impression of what Ibis has to offer
  - Never complete – focus on:
    - most important lessons learned
    - best match with ‘you’

Disclaimer #3

- Ibis is not the-be-all-and-end-all solution
  - A modular toolbox
    - Many (integrated) solutions to fundamental problems
  - Some Ibis-tools are low-level
    - but higher level APIs, models, GUIs often available
  - Further research required
  - Further development / engineering required
- Note:
  - Code itself (in principle) less important than the knowledge it contains

Disclaimer #4

- Ibis does not (explicitly) provide solutions for
  - World peace, cold fusion, stock market prediction, … ☺
  - …
  - Security
    - Well, there is no Ibis-specific security
    - Do integrate existing solutions, e.g. ssh tunnels, grid certificates
  - Co-allocation
    - Well, there is a JavaGAT call, but adaptors must implement it
    - Do provide support for malleability and migration
- It's all a matter of 'scope':
  - Please help us define this properly

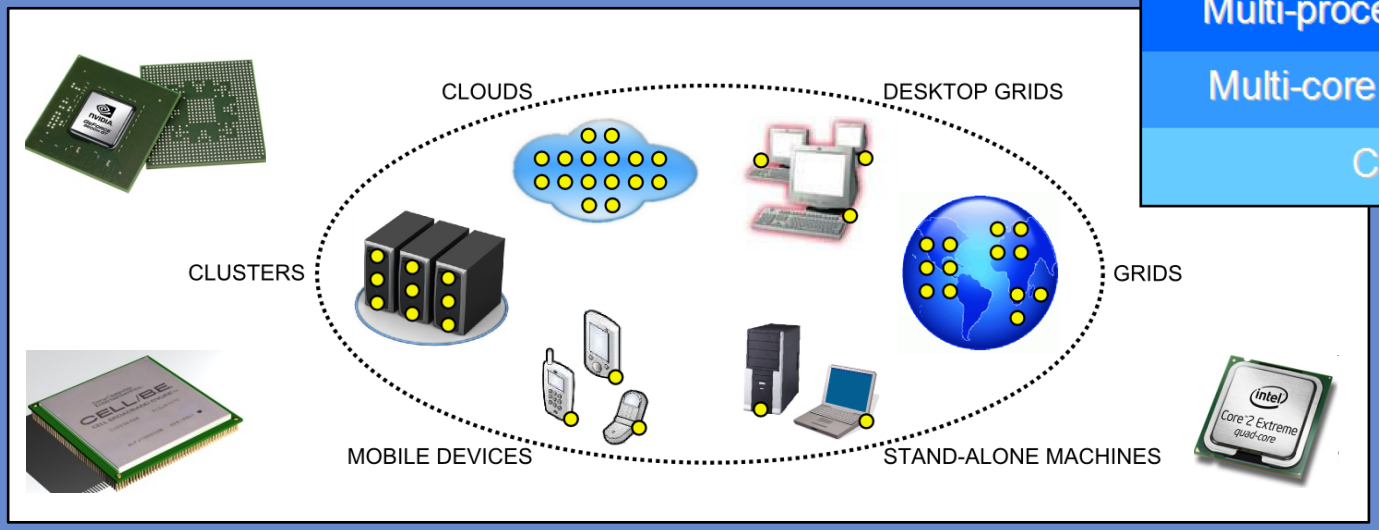
Disclaimer #5

- Outreach to Application Domains
  - We are happy to support any user, but time is against us
    - We must do research primarily
    - Currently we do ‘cherry-picking’
    - Future: through NLeSC, or…?
- Outreach to Expert HPDC Developers
  - Ibis is open source; contributions always welcome
    - So far: mostly JavaGAT adaptors (e.g. from D-Grid)
    - Future: through NLeSC, or…?
  - Roadmap to ‘Ibis Coding Community’ required

Jungle Computing

Jungle Computing

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

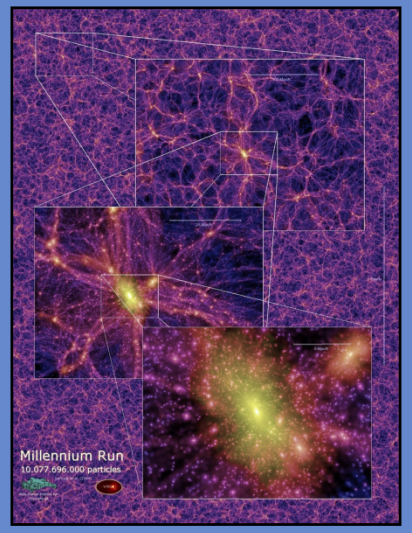


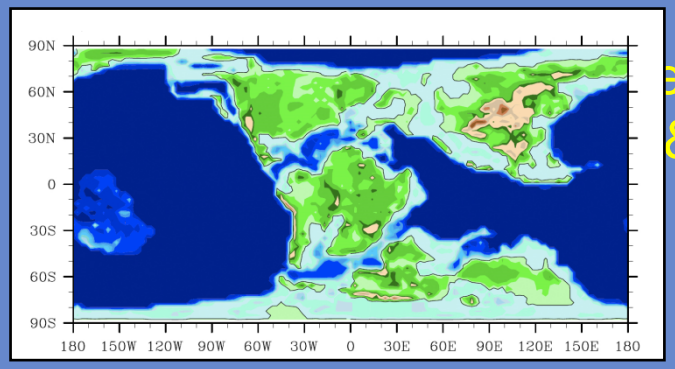
Why Jungle Computing?

- 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

Example Application Domains

- Computational Astrophysics (Leiden)
  - AMUSE: multi-model / multi-kernel simulations
  - “Simulating the Universe on an Intercontinental Grid” - Portegies Zwart et al (IEEE Computer, Aug 2010)
- Climate Modeling (Utrecht)
  - CPL: multi-model / multi-kernel simulations
    - Atmosphere, ocean, source rock formation, …





erse
& scalability
- …

- hardware:
  - high

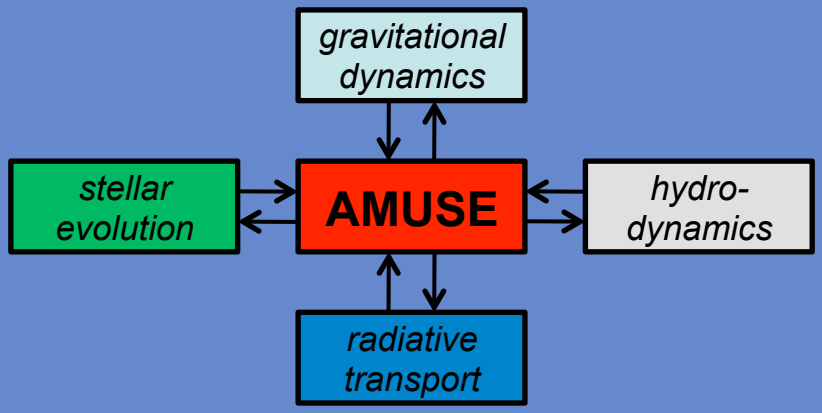
Domain Example #1:
Computational Astrophysics

Domain Example #1: Computational Astrophysics

Demonstrated live at SC'11, Nov 12-18, 2011, Seattle, USA (two week ago)

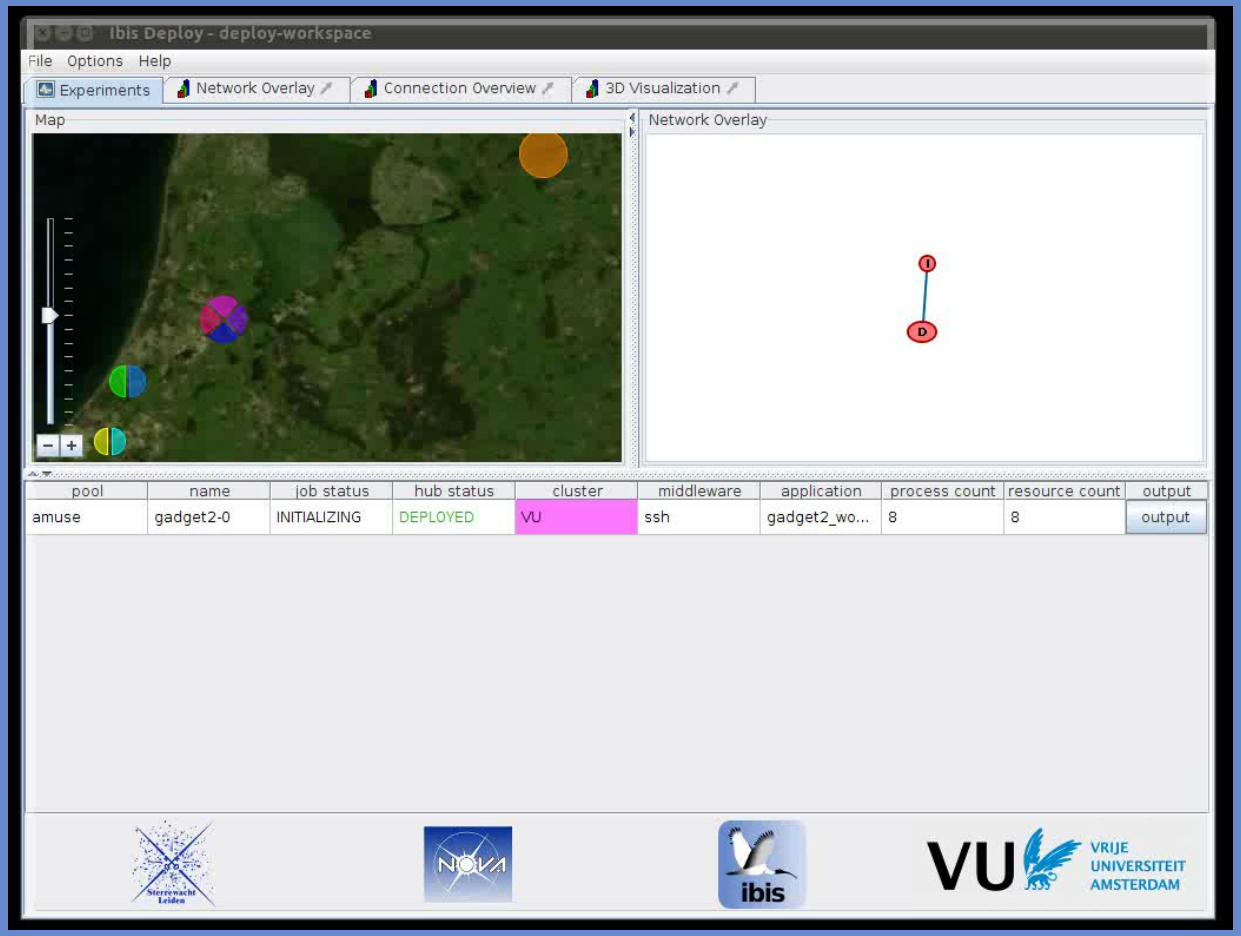
Domain Example #1: Computational Astrophysics

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



- 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 (two weeks ago)

Domain Example #2:
Multimedia Content Analysis

Multimedia Content Analysis (MMCA)

- Aim:
  - Automatic extraction of ‘semantic concepts’ from image sets and video streams

  - Depending on specific problem & size of data set:
    - May take hours, days, weeks, months, years…

Multimedia Content Analysis (MMCA)

- Need for user-friendly programming tools
  - Shield domain-experts from *all* complexities of parallel, distributed, heterogeneous, and hierarchical computing
    - Familiar (sequential) programming model(s)

User

Solution:
tool to make parallel & distributed computing transparent to user

- familiar programming
- easy execution

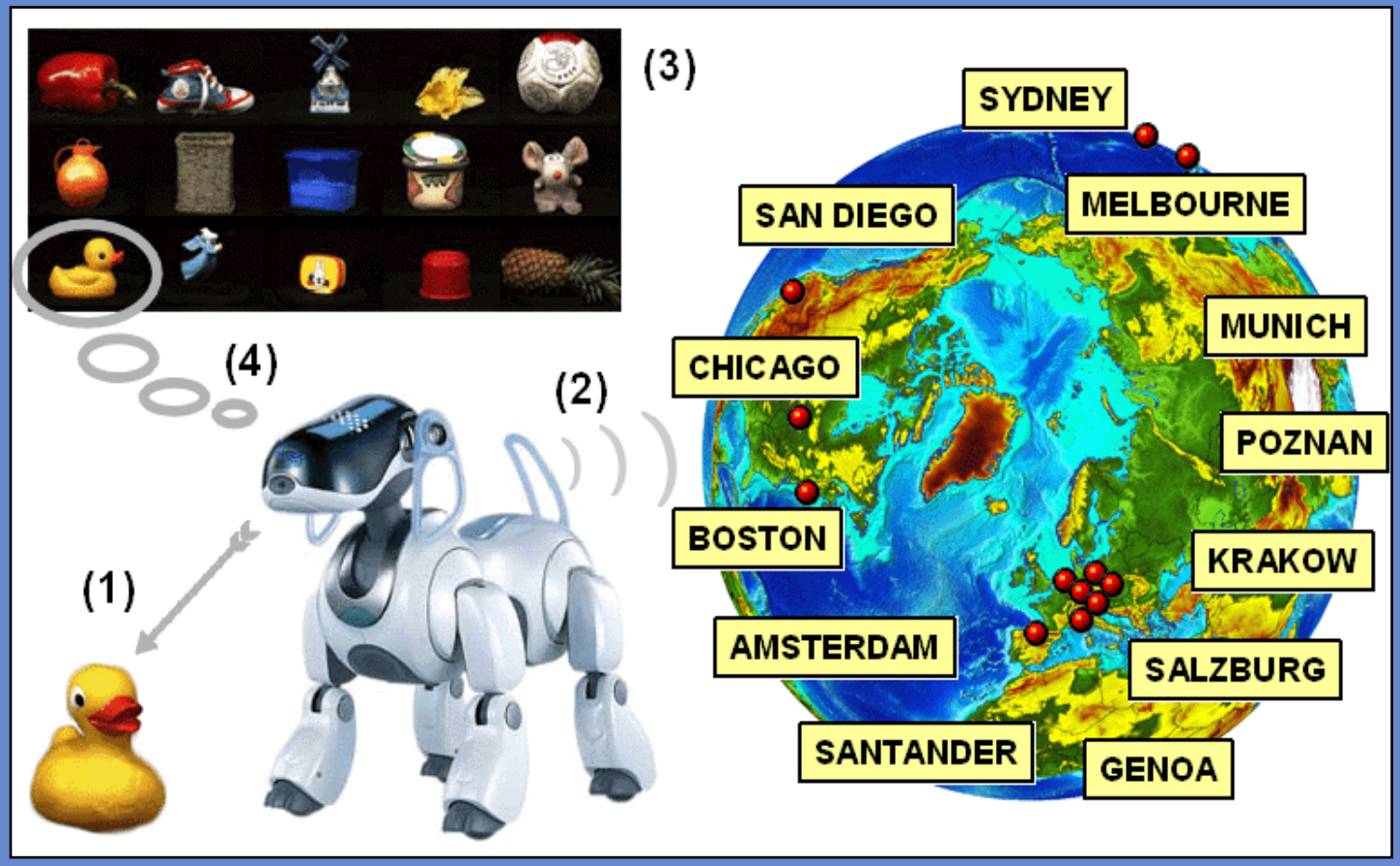
Jungle Computing Systems

Multimedia Content Analysis (MMCA)

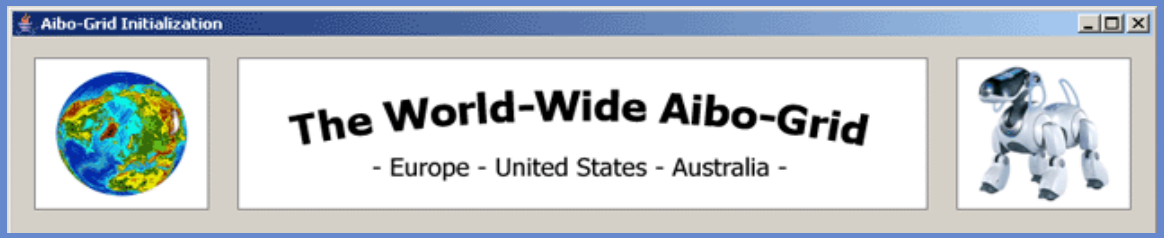
- Applications in (a.o):
  - Remote Sensing
  - Security / Surveillance
  - Medical Imaging
  - Document Analysis
  - Multimedia Systems
  - Astronomy

- Application types:
  - Real-time vs. off-line
  - Fine-grained vs. coarse-grained
  - Data-intensive / compute-intensive / information-intensive

Domain Example #2: Color-based Object Recognition by a Grid-connected Robot Dog



Seinstra et al (IEEE Multimedia, Oct-Dec 2007)
Seinstra et al (AAAI'07: Most Visionary Research Award)


Aibo-Grid Initialization
The World-Wide Aibo-Grid
- Europe - United States - Australia -

Successful…

- …but many fundamental problem unsolved!
  - Scaling up to very large systems
  - Platform independence
  - Middleware independence
  - Connectivity (a.o. firewalls, …)
  - Fault-tolerance
  - …
- Software support tool(s) urgently needed!
  - Jungle-aware + transparent + efficient
  - No progress until ‘discovery’ of Ibis

The Ibis Software Framework

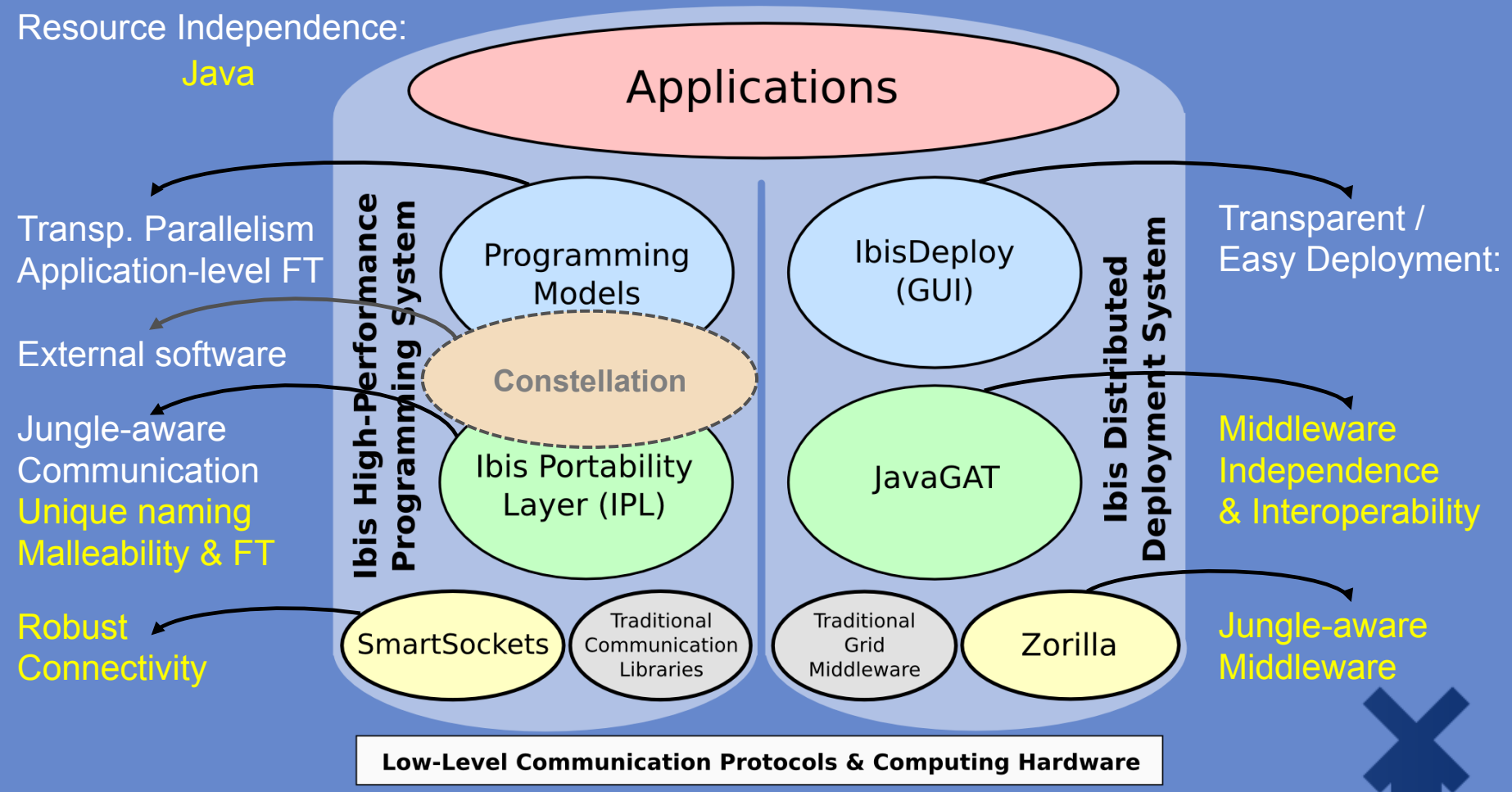
The Ibis Software Framework

- Offers all functionality to efficiently & transparently implement & run Jungle Computing applications
- Designed for dynamic / hostile environments
- Modular and flexible
  - Allow replacement of Ibis components by external ones, including native code
- Open source
  - Download: http://www.cs.vu.nl/ibis/

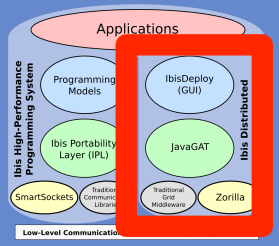
General Requirements

- Resource independence
- Transparent / easy deployment
  - Middleware independence & interoperability
  - Jungle-aware middleware
- Jungle-aware communication
  - Robust connectivity
  - System-support for malleability and fault-tolerance
  - Globally unique naming
- Transparent parallelism & application-level fault-tolerance
- Easy integration with external software (legacy codes)
  - MPI, CUDA, C++, Python, Java, Fortran, …

Ibis Software Stack

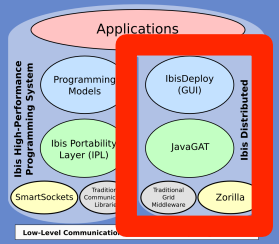


JavaGAT

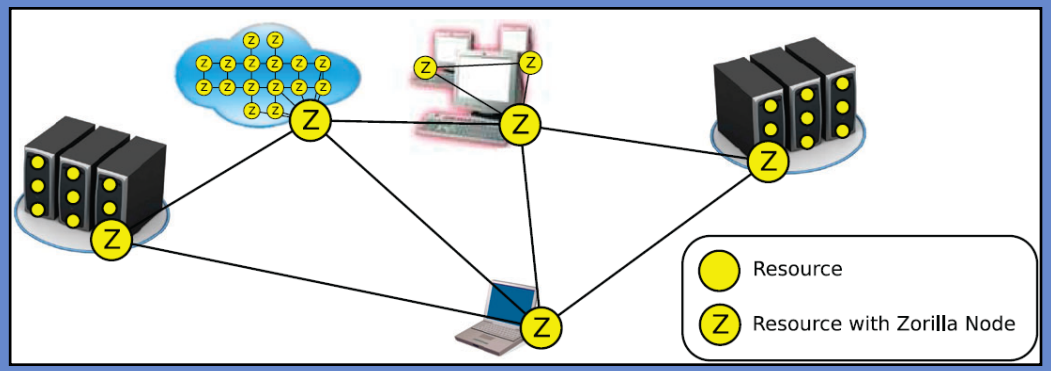


- Java *Grid* Application Toolkit
  - High-level API for developing (*Grid*) applications *independently* of the underlying (*Grid*) middleware
  - Use (*Grid*) services; file cp, resource discovery, job submission, …
  - Overcomes problems, incl:
    - Functionality may not work on all sites, or for all users, …
    - Middleware version differences & complex codes…
- Note: SAGA API standardized by OGF
  - Simple API for *Grid* Applications (a.o. with LSU)
  - SAGA on top of JavaGAT (and v.v.)

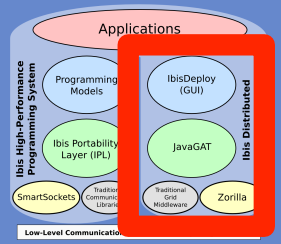
Zorilla (not discussed; future)

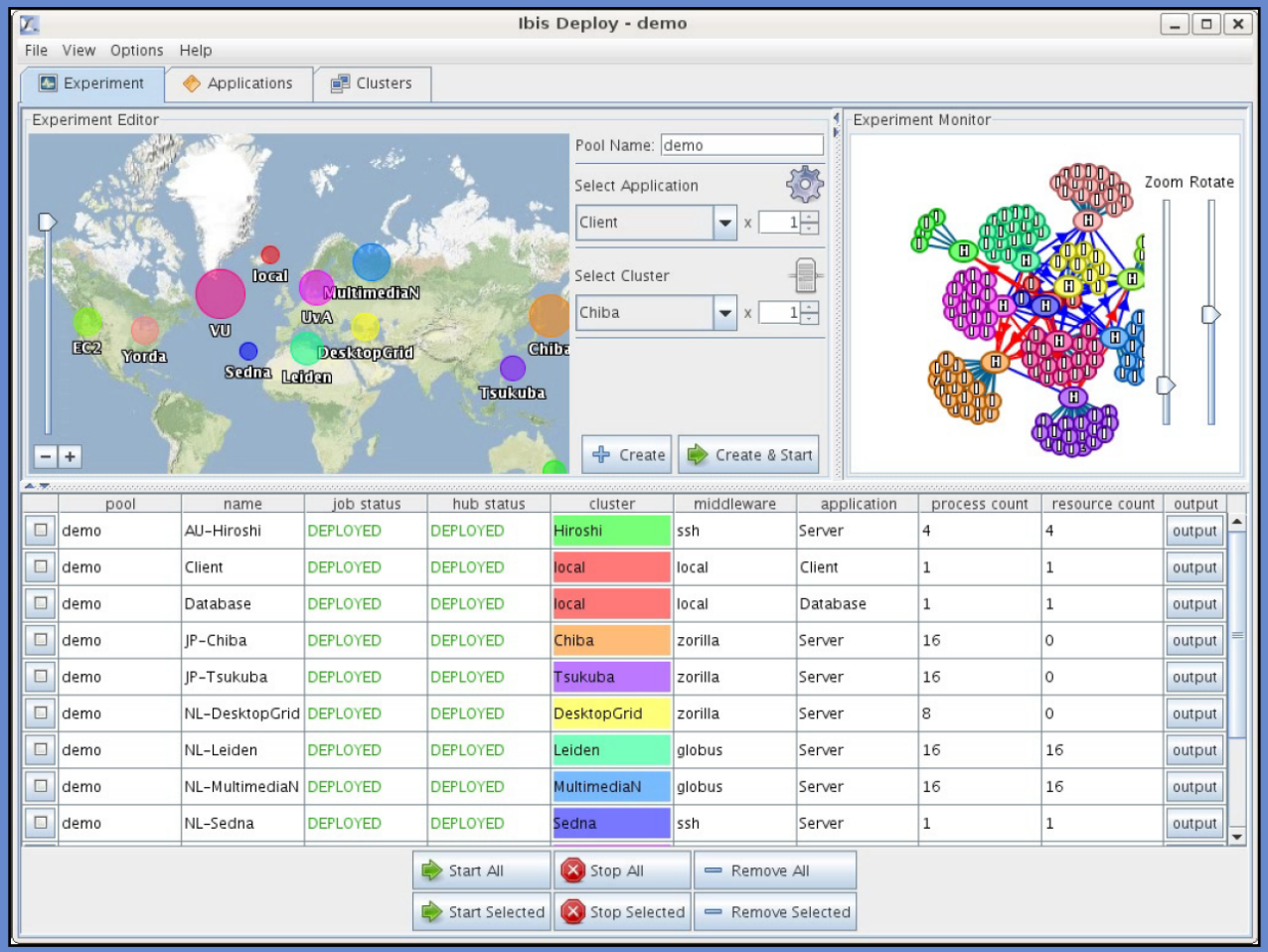


- A prototype P2P middleware
  - A Zorilla system consists of a collection of nodes, connected by a P2P network
  - Each node independent & implements all middleware functionality
  - No central components
  - Supports fault-tolerance and malleability
  - Easily combines resources in multiple administrative domains



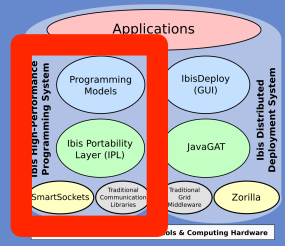
IbisDeploy





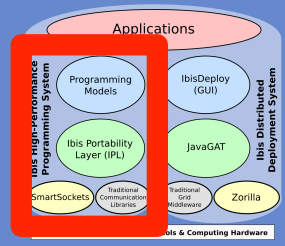
pool	name	job status	hub status	cluster	middleware	application	process count	resource count	output
demo	AU-Hiroshi	DEPLOYED	DEPLOYED	Hiroshi	ssh	Server	4	4	output
demo	Client	DEPLOYED	DEPLOYED	local	local	Client	1	1	output
demo	Database	DEPLOYED	DEPLOYED	local	local	Database	1	1	output
demo	JP-Chiba	DEPLOYED	DEPLOYED	Chiba	zorilla	Server	16	0	output
demo	JP-Tsukuba	DEPLOYED	DEPLOYED	Tsukuba	zorilla	Server	16	0	output
demo	NL-DesktopGrid	DEPLOYED	DEPLOYED	DesktopGrid	zorilla	Server	8	0	output
demo	NL-Leiden	DEPLOYED	DEPLOYED	Leiden	globus	Server	16	16	output
demo	NL-MultimediaN	DEPLOYED	DEPLOYED	MultimediaN	globus	Server	16	16	output
demo	NL-Sedna	DEPLOYED	DEPLOYED	Sedna	ssh	Server	1	1	output

Ibis Portability Layer (IPL)

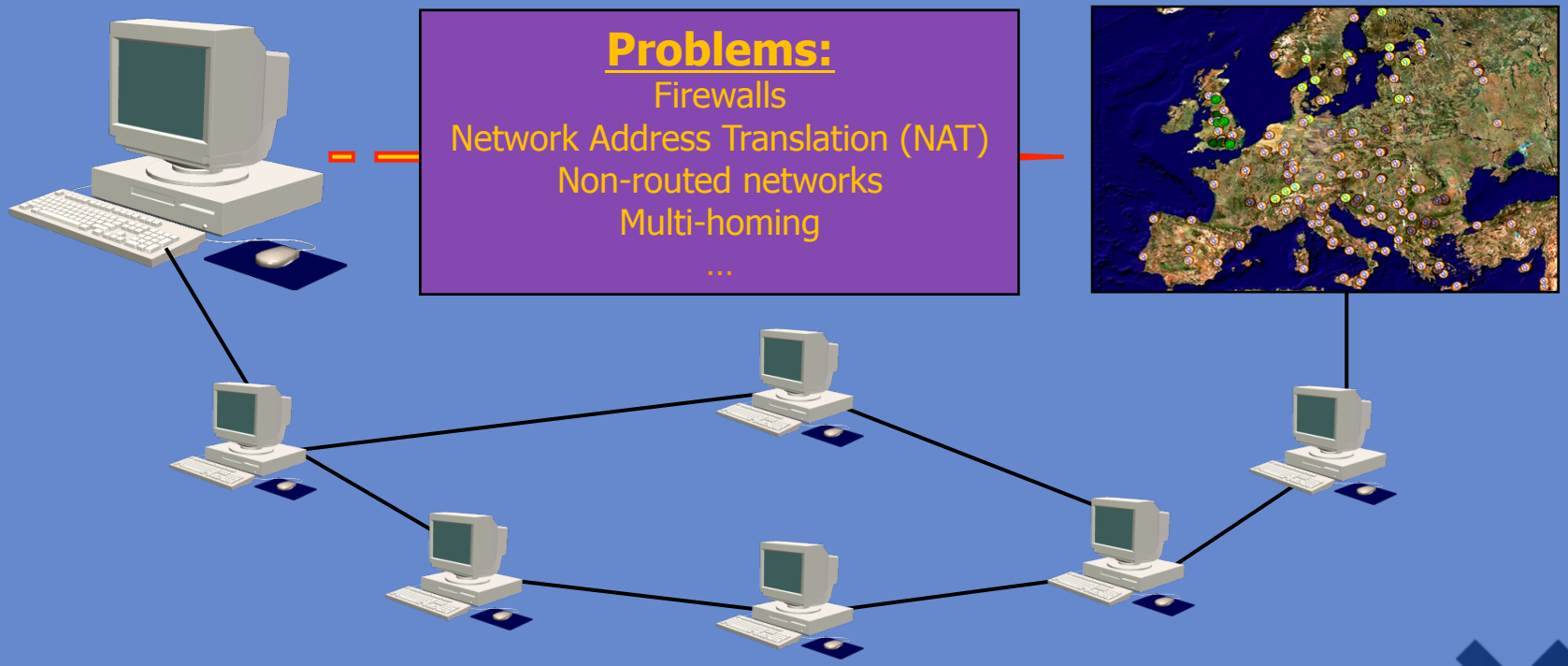


- Java-centric ‘run-anywhere’ communication library
  - Sent along with your application
  - “MPI for the Grid” (quote 2005)
- Supports fault-tolerance and malleability
  - Resource tracking (JEL model)
  - Open-world / Closed world
- Efficient
  - Highly optimized object serialization
  - Can use optimized native libraries (e.g. MPI, MX)

SmartSockets

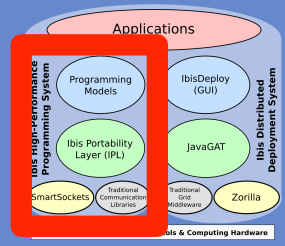


- Robust connection setup

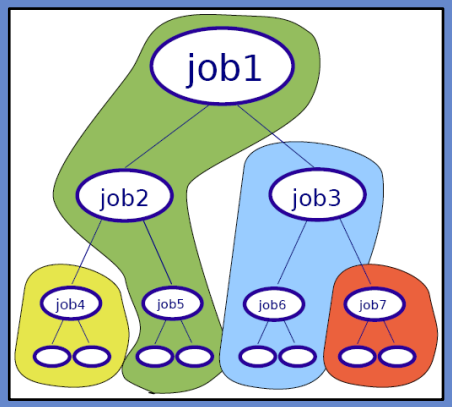


- Always connection in 30 different scenarios

Ibis Programming Models

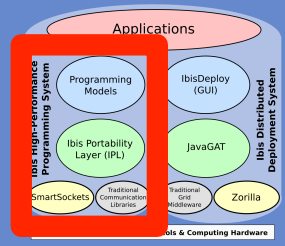


- IPL-based programming models, a.o.:
  - Satin:
    - A divide-and-conquer model
  - MPJ:
    - The MPI binding for Java
  - RMI:
    - Object-Oriented remote Procedure Call
  - Jorus:
    - A ‘user transparent’ parallel model for multimedia applications

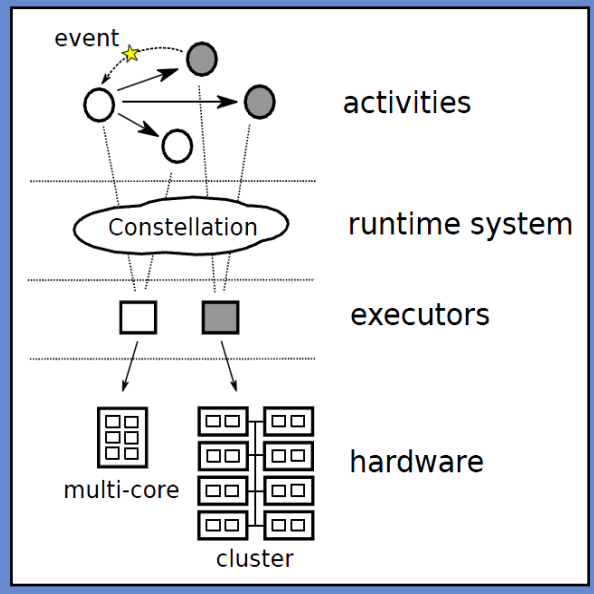


- Note:
  - In most cases intended as ‘proof-of-concept’ (= scientific)
  - Not discussed in this mini-course

“The Future of Ibis”



- Ibis/Constellation:
  - Generalized programming framework for ‘all’ Jungle Computing applications
  - Automatically maps *any* application activity (task) onto *any* appropriate executor (HW)
  - By way of ‘contexts’, for example:
    - Activity's context: “I need a GPU”
    - Executor’s context: “I represent a GPU”



- Note:
  - Activities may represent any type of task:
    - Also legacy codes, scripts, 3rd party software, …

The 3 Common Uses of Ibis

Ibis as ‘Master Key’ (or ‘Passepartout’)

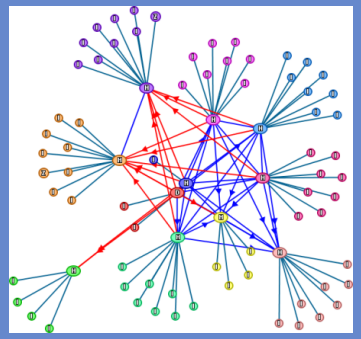
- Use JavaGAT to access ‘any’ system
  - Develop/run applications *independently* of available middlewares
  - JavaGAT ‘adaptors’ required for each middleware
  - ‘Intelligent dispatching’ even allows for transparent use of *multiple* middlewares
- Example: file copy
  - JavaGAT vs. Globus

  - Simple, portable, …
  - SAGA API standardized

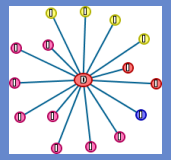
Ibis as 'Glue'

- 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:

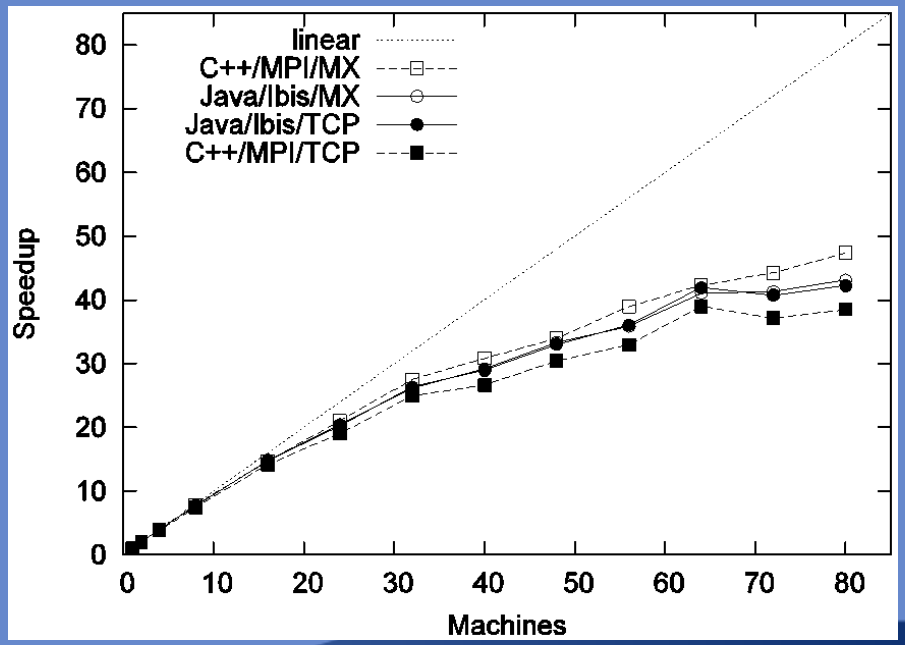


No SmartSockets:

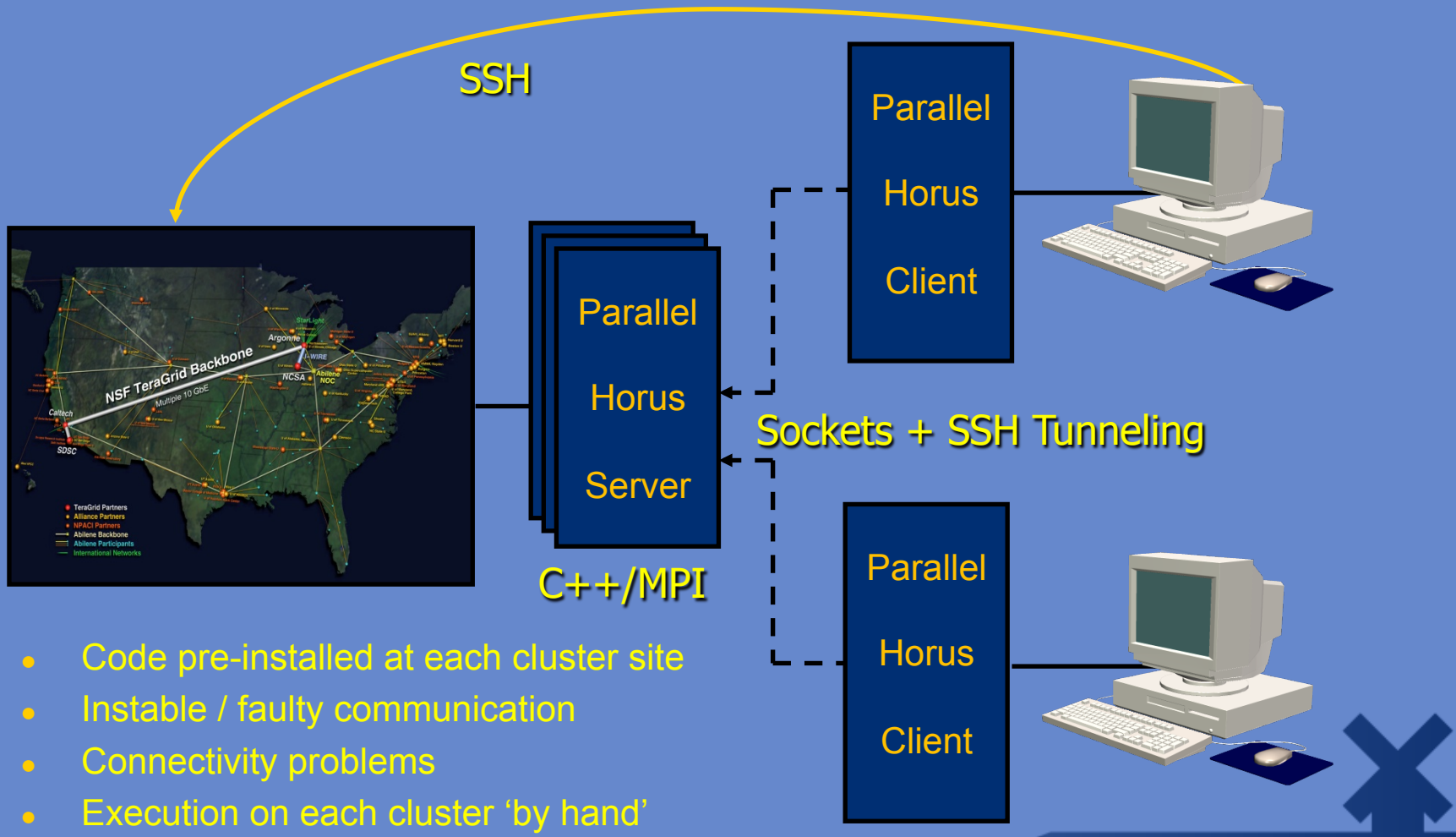


Ibis as 'HPC Solution'

- 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:
- Not discussed here

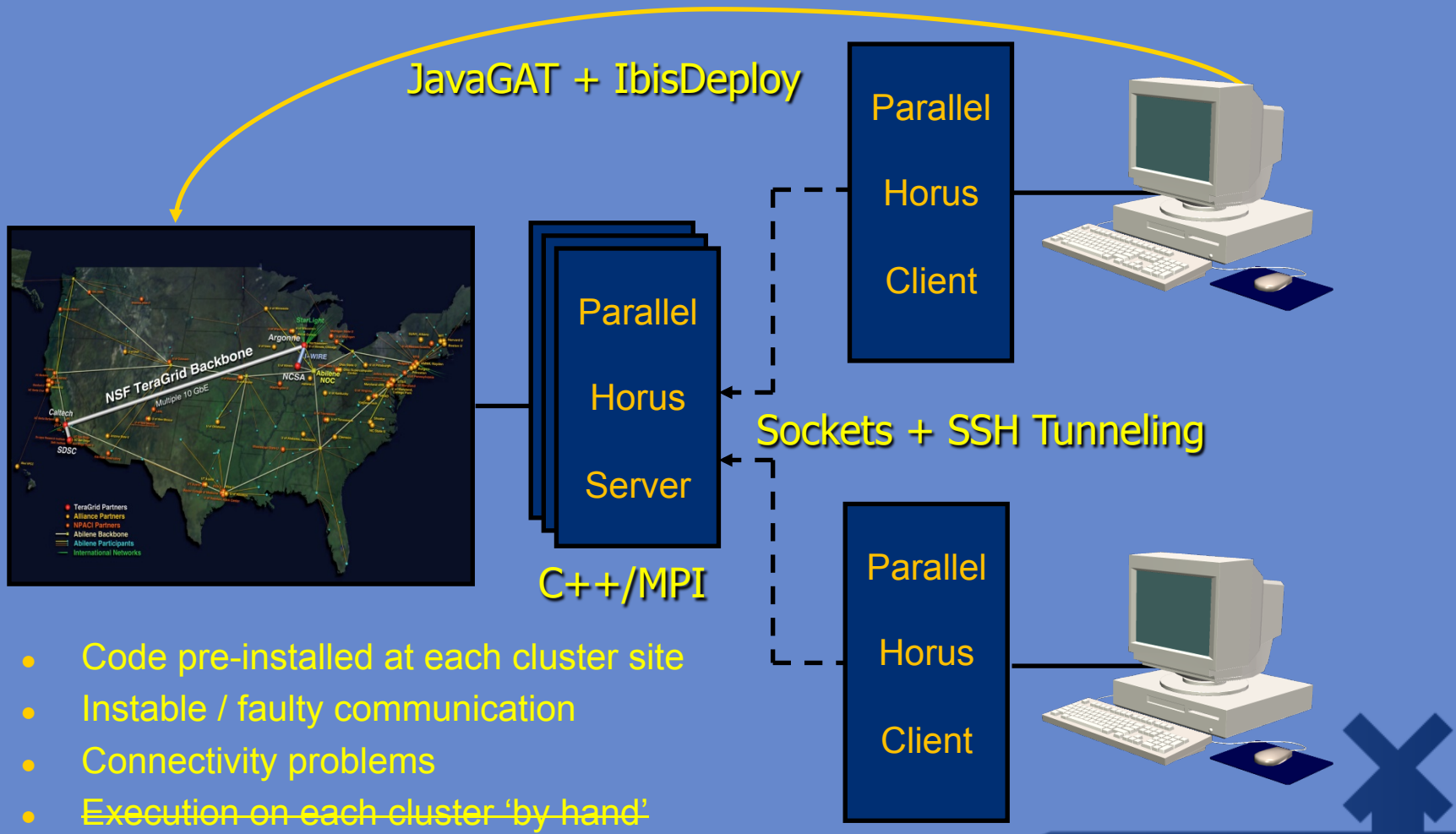


MMCA: Situation in 2004/2005



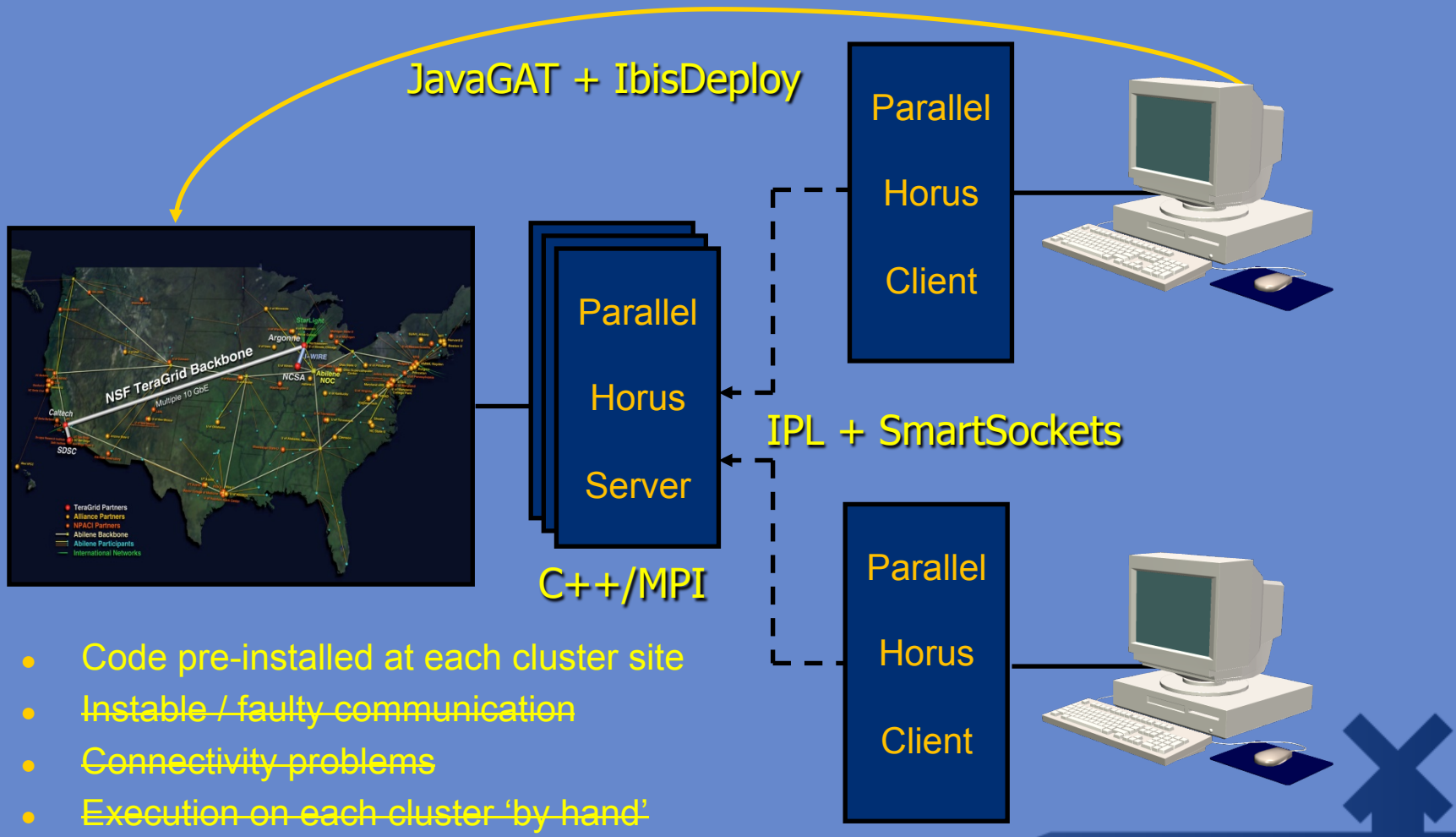
- Code pre-installed at each cluster site
- Instable / faulty communication
- Connectivity problems
- Execution on each cluster 'by hand'

Phase 1: Ibis as 'Master Key' (2006)



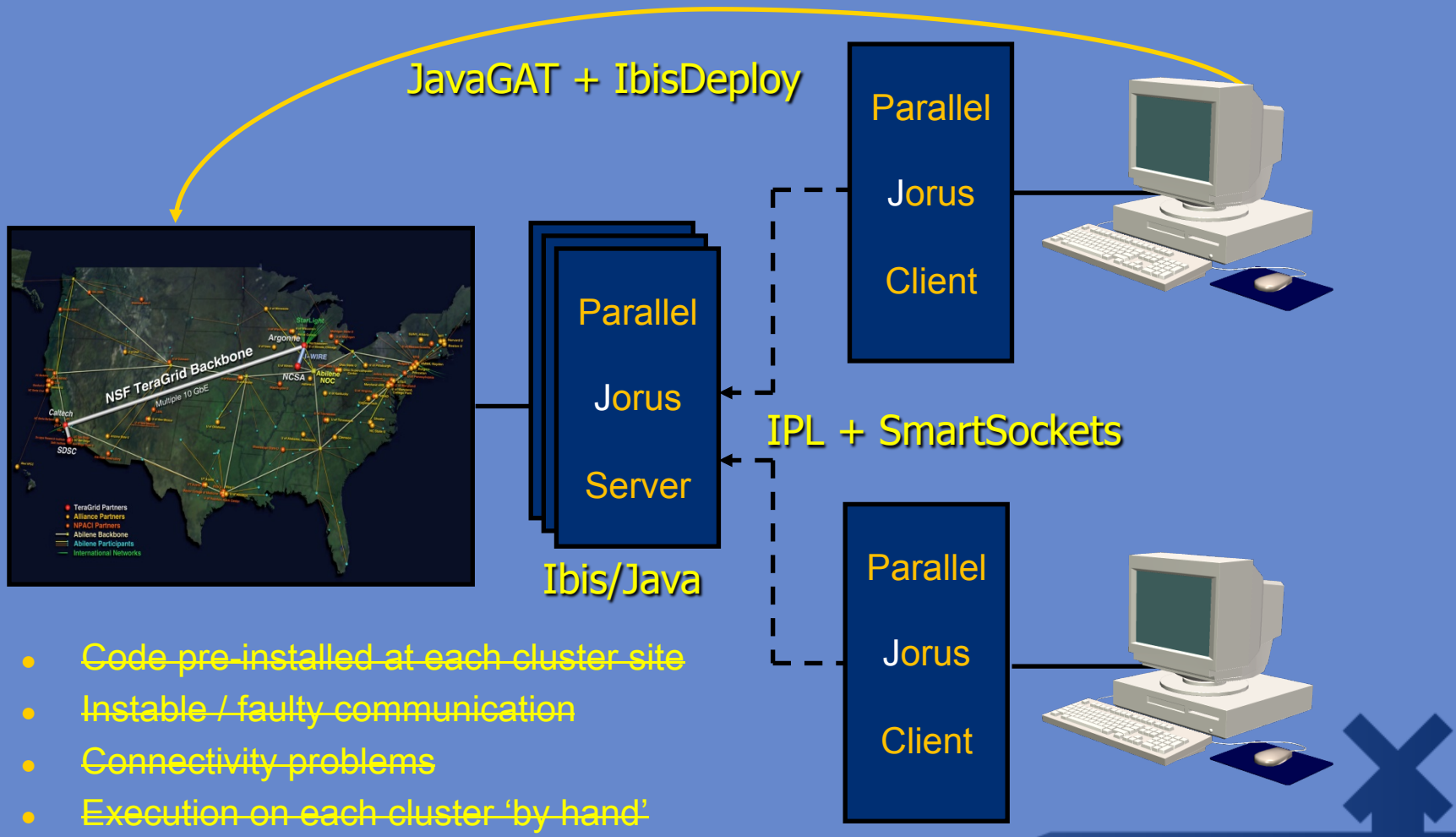
- Code pre-installed at each cluster site
- Instable / faulty communication
- Connectivity problems
- ~~Execution on each cluster 'by hand'~~

Phase 2: Ibis as 'Glue' (2006/2007)



- Code pre-installed at each cluster site
- ~~Instable / faulty communication~~
- ~~Connectivity problems~~
- ~~Execution on each cluster 'by hand'~~

Phase 3: Ibis as 'HPC Solution' (2008)



- ~~Code pre-installed at each cluster site~~
- ~~Instable / faulty communication~~
- ~~Connectivity problems~~
- ~~Execution on each cluster 'by hand'~~

‘Master Key’ + ‘Glue’ + ‘HPC’

- 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’
  - After each phase a fully functional, working solution was available!
- Eventual result:
  - ‘wall-socket computing from a memory stick’
  - Remember: the ‘Promise of the Grid’?
  - Awards at AAAI 2007 and CCGrid 2008

100% Ibis Implementation (2008++)

Seinstra, Maassen, Drost et al. (SCALE 2008 @ CCGrid 2008: First Prize Winner)
Bal, Maassen, Drost, Seinstra et al. (IEEE Computer, Aug 2010)

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/

Conclusions (2)

- Jungle Computing is hard
- High-Performance Jungle Computing even harder
- While research into efficient & transparent Jungle-aware programming models has only just begun…
- …Ibis provides the basic functionality to efficiently & transparently overcome most Jungle Computing complexities

End of Introduction

www.cs.vu.nl/ibis/