Grid Programming Models: Current Tools, Issues and Directions

Craig Lee

Computer Systems Research Department
The Aerospace Corporation, P.O. Box 92957
El Segundo, CA USA
lee@aero.org

Domenico Talia

DEIS
Universitá della Calabria
87036 Rende, CS Italy
talia@deis.unical.it

Ole Weidner

oweidner@cct.lsu.edu CCT, LSU, Baton Rouge
What is a PM (i)

- Not clearly defined - different groups have different notions
- Abstract conceptual view of the structure and operation of a system
- Usually bound to a specific application domain
- Must be applicable to the real world - the PM’s implementation.
“The object of a programming model is to make a certain class of problems easier to solve, perhaps at the risk of making others more difficult. As a rule, some problems are solved naturally in a given model, while others are not. Echoing the age-old wisdom of selecting the right computer language for the task, this is even more critical when the tool is made more specific.“

- Lewis Girod UCLA/MIT
Why use a PM

- Abstraction (hopefully) simplifies a domain-specific development process
- Provides a theoretical discussion basis for computer scientists
- Provides (industry) standard requirements that different implementations can adopt
Some example PMs

- W3C Web Services (SOAP)
- Thread Programming Model
- OpenGL (debatable)
- Message Passing Interface (MPI)
- CORBA
- SGI SHMEM™
- Charm++
The Outline

• Identify requirements for Grid programming models and tools
• Overview of existing models, implementations & tools in different categories
• Discussion of current weaknesses and possibilities of improvement
To identify a set of requirements we need to know the issues in Grid Programming:

Everything from Parallel Programming...
Requirements (ii)

- **Portability**
  - Grids are dynamic and heterogeneous
  - Code should be architecture independent
    - different sets of pre-staged codes
  - interpreted virtual machine
  - *PM implementations should support the development of portable code*
Requirements (iii)

- **Interoperability**
  - Codes can use different implementations of a PM (e.g. vendor-specific MPI)
  - *Different implementations of a PM should be equivalent*
  - Open protocols, services and interfaces are crucial to support that
Requirements (iv)

- **Resource Discovery**
  - Grids can be unknown and/or volatile
  - Grids should advertise their services in a uniform way so that applications can dynamically discover them
  - *Grid PMs should incorporate resource discovery as a fundamental concept*
Requirements (v)

- **Performance**
  - Bandwidth and latency are heterogeneous which can result in widely varying performance results
  - Many applications need a deterministic performance model
  - *Introduction of QoS can provide a reliable performance forecast*

CSC7700 Grid Computing
Requirements (vi)

- **Fault Tolerance**
  - Highly distributed codes can run jobs on thousands of hosts
  - Applications should always assure a minimum level of reliable computation
  - BUT: Grid PM should provide run-time error checking and actions to react / recover
Requirements (vi)

- **Security**
  - Grid codes run on shared resources across multiple administrative domains
  - *Grid PMs should provide mechanisms for*
    - **Authentication**
    - **Privacy (e.g. encryption)**
Req. Conclusion

- We’ve discovered 6 requirements for PMs:
  - Portability
  - Interoperability
  - Resource Discovery
  - Performance
  - Fault Tolerance
  - Security
Tools & Models (i)

• Shared-State Models
  - Usually found in tightly-coupled systems (shared address space)

• JavaSpaces
  - Applications communicate via shared objects
  - Implements distributed persistent objects (Linda tuple-space) “spaces”
  - Implementation on top of Globus
Tools & Models (ii)

- Message-Passing Models
  - Processes in disjoint address spaces using messaging for communication
  - MPI: Wide-spread but low-level
- MPICH-G2
  - Dynamically switches transfer protocols
  - Grid-aware version Implemented on top of Globus
RPC and RMI Models

- Client/Server architecture
- Allows to trigger functions on remote machines
- **Grid-enabled RPC (GridRPC)**
  - Supports resource discovery
  - Supports security (X.509/GSI)
  - Supports fault tolerance
- Prototypes exist on Ninf and NetSolve/GridSolve
Tools & Models (iv)

- RPC and RMI Models (cont.)
  - **Java RMI**
    - Call methods across VM boundaries
    - Can be used to write distributed objects
    - Java provides portability and a security model
    - **BUT**: No other Grid issues addressed
Tools & Models (vi)

- Hybrid Models
  - Combining models to serve multiple domains (e.g. Parallel Comp. PM + Grid PM)
  - **OpenMP/MPI** (SMP + MPI)
  - **OmniRPC** (OpenMP + RPC)
  - **MPJ** (SMP + RMI + MPI)
Tools & Models (vii)

- Peer-to-Peer (P2P) Models
  - Decentralized self-organizing peer networks
- JXTA (Juxtapose)
  - P2P XML protocol specification by SUN
  - Allows advertise and discover resources
  - Form/join resource groups
  - “Ad-hoc Grids”
Tools & Models (viii)

- Frameworks, Comp. Models, APIs, Portals
  - Cactus
  - CORBA
  - CoG Kit
  - Legion
  - GridSphere
  - GAT / SAGA
Tools & Models (ix)

- Web Service Models
  - Similar to RPC (HTTP-based)
  - Quite hyped industry standard ;-)
- OGF OGSA
  - Specifies service-oriented grid architecture: data handling, jobs, security, inform. services
  - Globus Toolkit 4 is an OGSA implementation
T & M Conclusion (i)

- Lots of traditional HPC Models & Tools were adapted to the Grid
- Many traditional PM’s were “gridified” using the Globus Toolkit. Why?
- Because Globus satisfies lots of the identified requirements
- Globus is de-facto standard for Grids and virtually available everywhere
• Some of the current weaknesses are
  • Virtually all models were derived from HPC no first principle Grid Programming Models
  • More work must be done in the fields of
    • data-driven & optimistic programming
    • advanced communication
    • I/O services
Finally

“Will computational science be limited to the size of single-chassis machines, such as the ASCI machines and the HTMT? Or can the problem architectures for science and engineering, and their associated computational models, be made sufficiently *grid-friendly* such that increasingly large problems can be solved? Much work remains to be done.”

- [Lee03]