Managed by

Report on required features to be included in Coregrid's Grid Component Model

Massimo Coppola

Dept. of Computer Science, Univ. of Pisa

and ISTI-CNR, Pisa

CoreGrid Virtual Institute on Programming Models

coppola@di.unipi.it

Overview

- Grid Component Model
- Requirements
  - Fundamental
  - Parallel and Distributed
  - Adaptivity
  - Heterogeneity and Deployment
- Multi-architecture perspective
- Conclusions

Motivation

Grid platforms

- Beyond parallel and distributed programming
- Dynamic, heterogeneous (SW, HW, MW…), pervasive
- Unreliable… but not even latest/largest MPP are!

Efficient programming tools *in-the-large*

- ***Programmability*, *interoperability, scalability***

CoreGrid roadmap : exploit components

- Application design and implementation
  - Component design
  - Composition
- Deployment
- Application management / steering (QoS, FT …)

…to manage Grid aspects that are orthogonal to applications!

GCM basics and roadmap

Abstract view

Maps to / is a superset of different instances of component models

- **Mapping to different languages**
- **High level XML descriptor**

What next: GRID aspects

- **Define types of ports**
- **External / internal interfaces**
- **"Standard" non-functional properties**

Develop run-time support

Activity

Fractal model as a reference

What Non-functional aspects are essential to Grids?

Gathering Additional Requirements

- **Several proposals already from WP3**
  - **INRIA - Rennes, INRIA -Sophia, Uni. Pisa, Uni. Muenster, Uni. Westm. …**
- **Additional ones from other WPs and projects**
  - **New Technical Working Group on Programming Models for the Grid**

Discussion and integration

Draft of Requirements for GCM

Requirements and Meta-requirements

Programmability

Interoperability

Extensibility

Fast Prototyping

Scalability

Efficient Management in-the-large

High Performance

Fundamental requirements (1)

Support for reflexivity

- Both *introspection* and *intercession*

Model Extensibility

- Placeholders in the model for adding features to components

Language neutrality

- **GCM should be implemented in a language-independent way**

Tradeoffs w.r.t. *lightweight*, homogeneous, *formal (means well defined semantics)*

Fundamentals (2) : *Packaging*

Packaging standards

- standard packaging description of components (e.g. jar files)

ADL with support for deployment

- A standard ADL descriptions of characteristics for distributed deployments
- **Virtual computing topology for component deployment**
  - Proactive Virtual Nodes
  - ASSIST Virtual Processes

Extensible packaging

- Extra information in packages (e.g. an adaptation policy)
- Meta-information or executable code?

Compositionality requirements

composition

Support for Higher-order component programming

skeleton-parallel, functional

- complex issue: how to exchange code / code carrying parameters
- what kind of code?
  Executables, VM bytecode, components/component references to be deployed?

Web Service *interoperability*

- Exporting WSDL services is a need for "industrial standard" compliance.
- How much WSDL compatibility must limit/influence the component model used to develop applications?

Parallelism

Sequential and parallel implementation.

- Support both
- **Support several alternatives (and associated meta-data…)**
- Example:
  - Java and Fortran sequential implementations
  - MPI and OpenMP parallel implementations
- Parallel components distributed over Grids
  - Inter-organizational collaboration (Virtual Organizations)
  - *What kind of problems at the next plugtest?*
- Impact on Deployment

Interfaces: asynchronous

Extended / Extensible port semantics

Less/no constraints on service provided by ports

- **interface oriented (e.g. provides/uses with synchronous, asynchronous, deferred mode; futures),**
- **message oriented (e.g. CCM events, ASSIST data-flow streams, data-space oriented)**
- **any still unknown way of communicating:**
  - **E.g. explicitly nondeterministic communication**

Parallelism and Communications (1)

Multiple/asymmetric port connection

- One port connected to many (e.g. CCM events)

Issuing and collecting parallel multiple calls

- A notion of parallel Server and Client interfaces, (calls being sent/received in parallel
- gathering calls together for the sake of synchronization
- Group-related communications on interfaces

Parallelism and Communications (2)

MxN communications

- Interface to mechanisms for achieving MxN redistribution

A standard mechanism / NF interface to

- **Negotiate communication protocols** at deploy time
- Renegotiate them dinamically

Parallel binding checking

- A binding that ensures some static checks on the correctness of binding parallel components together

Adaptivity

Extension of the process of building a component.

Adding new roles

- **"adaptation expert" to devise adaptation policies**

Adding new phases

- **aspect weaving, automated transformation**

Extensibility of the design process of components:

EITHER allow unanticipated extensions of the model

OR give a standard for adaptive behavior

We're not making assumptions on the kind of code within a component…

Define component interfaces for aspects?

Adaptivity (2)

Adaptivity implementation within components

- Ability to plug/unplug components dynamically

Dynamic Reconf. of Grid-distrib. App.s. Who's in control?

Exploit Component Hierarchical abstraction for adaptivity

- Components support a local adaptivity manager
  - Component-level adaptation
- Component "managers" need to interact with each other
  - Special kind of manager subcomponent
  - External/internal interfaces for configuration
  - External/internal interfaces for monitoring
  - Interaction of configuration and deployment

Need to plug/unplug Non-functional interfaces

Separation of concerns and aspects

Same hierarchical approach to deal with other Grid-related aspects and kinds of non-functional behavior

Service Level Agreement

- **QoS, FT, Performance Non-functional properties**
  - **Means to specify contracts**
  - **Monitoring/triggering interfaces**
  - **Provide interfaces for contract management**
    - **Manual steering, autonomic …**
- **Security level**
  - **A property of the infrastructure?**

Deployment

Deployment standards

- A standard description of deployment mechanisms

ADL with support for deployment

In-depth resource description

- component description to include details on
- middleware, App. SW, HW needed to run *all parts* of the component (see multimiddleware components)
- logic resources : files, I/O streams
- group and unitary requirements
- Deployment tied to program structure, NOT to app. code

Multi-middleware deployment

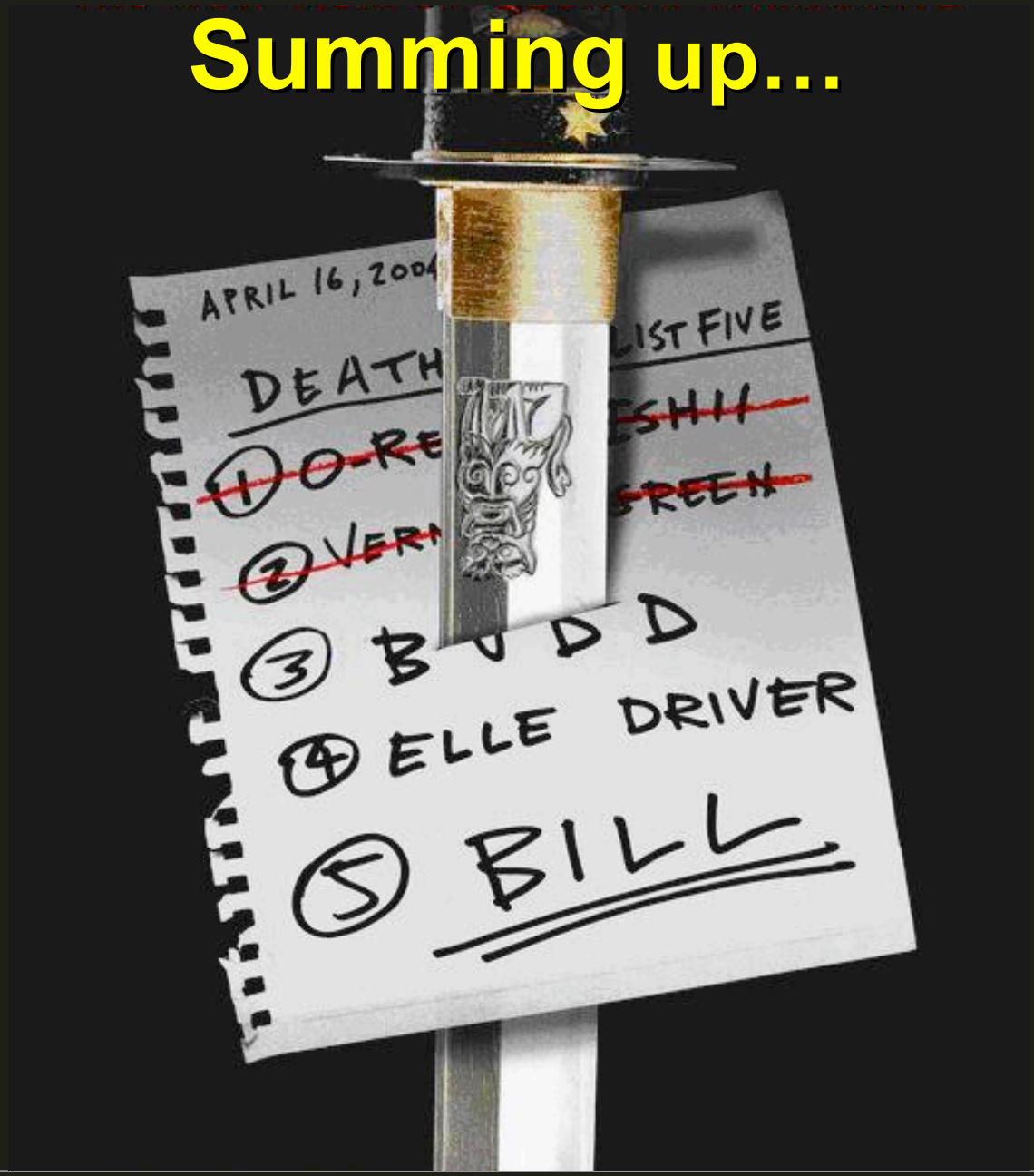
multi-middleware ports negotiating protocol at run-time

What's not been asked

Things in our agenda
but that were not explicitly mentioned

- **Specific mechanisms for Fault Tolerance**
  - **Checkpointing? FT communication handling?**
- **Specific mechanisms for security**
  - **Do we rely only on secure connections?**
  - **Is it below the horizon of the GCM?**
- **Component-level error handling**

Summing up…



Conclusions?

Almost any feature listed or required is related to research work ongoing

Need to actively merge different perspectives

Need to get more feedback:

within CoreGrid!

TWG on Programming models

Some Open Questions

Already a lot of features for a Lightweight / Extensible component model

- Is it a contradiction?
- Different Conformance levels?
- Identify the core features…

How to move code (adaptivity, higher order)

- Relying on the VM approach?
- Relying on the virtual resource approach?

Allow Grid aspects within the GCM

- Can we really merge aspects and components?
- Can we decide how different aspects interact?