

Resource Information and Description

- Last Time: Applications
 - » High Energy Physics – Massive Data Analysis and Simulation
 - » Encyclopedia of Life: Massive Computation
 - » LEAD: Linked Environments for Atmospheric Discovery
 - » NEESGrid: Hybrid Integrated Simulation
- Today
 - » Basics of Grid Resource Information Systems
 - » Resource Selection Languages
- Reminders/Announcements
 - » Nut Taesombut Lecture on Resource Selection on 4/12

CSE225 – Lecture #4

Today's Readings, Part I

- K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman, Grid Information Services for Distributed Resource Sharing, Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.
- A Resource Management Architecture for Metacomputing Systems. K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, S. Tuecke. Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing, pg. 62-82, 1998.
- Rajesh Raman, Miron Livny, and Marvin Solomon, "Policy Driven Heterogeneous Resource Co-Allocation with Gangmatching", Proceedings of the Twelfth IEEE International Symposium on High-Performance Distributed Computing, June, 2003
- Chuang Liu and Ian Foster, A Constraint Language Approach to Grid Resource Selection, University of Chicago, Department of Computer Science, TR-2003-07.
- David Oppenheimer, Jeannie Albrecht, David Patterson, and Amin Vahdat, Scalable Wide-Area Resource Discovery, UC Berkeley Technical Report UCB/CSD-04-1334, July 2004.
- Yang-Suk Kee, Dionysios Logothetis, Richard Huang, Henri Casanova, and Andrew A. Chien, Efficient Resource Description and High Quality Selection for Virtual Grids, In Proceedings of the IEEE Conference on Cluster Computing and the Grid (CCGrid 2005).

CSE225 – Lecture #4

Today's Readings, Part II

- Building Self-configuring Services Using Service-specific Knowledge, An-Cheng Huang and Peter Steenkiste. The Thirteenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-13), Honolulu, HI, June 4-6, 2004.
- T. Kichkaylo and V. Karamcheti, Optimal Resource-Aware Deployment Planning for Component-based Distributed Applications, Proceedings of the International Symposium on High Performance Distributed Computing (HPDC), July 2004
- ...but these cover Tuesday, April 12 Lecture as well!

CSE225 – Lecture #4

Resource Information Systems



Grid Information Services

- System information is critical to operation of the grid and construction of applications
 - What resources are available?
 - > Resource discovery
 - What is the “state” of the grid?
 - > Resource selection
 - How to optimize resource use
 - > Application configuration and adaptation?
- We need a general information infrastructure to answer these questions



Examples of Useful Information

- Characteristics of a compute resource
 - IP address, software available, system administrator, networks connected to, OS version, load
- Characteristics of a network
 - Bandwidth and latency, protocols, logical topology
- Characteristics of the Globus infrastructure
 - Hosts, resource managers



Grid Information: Facts of Life

- Information is always old
 - Time of flight, changing system state
 - Need to provide quality metrics
- Distributed state hard to obtain
 - Complexity of global snapshot
- Component will fail
- Scalability and overhead
- Many different usage scenarios
 - Heterogeneous policy, different information organizations, etc.

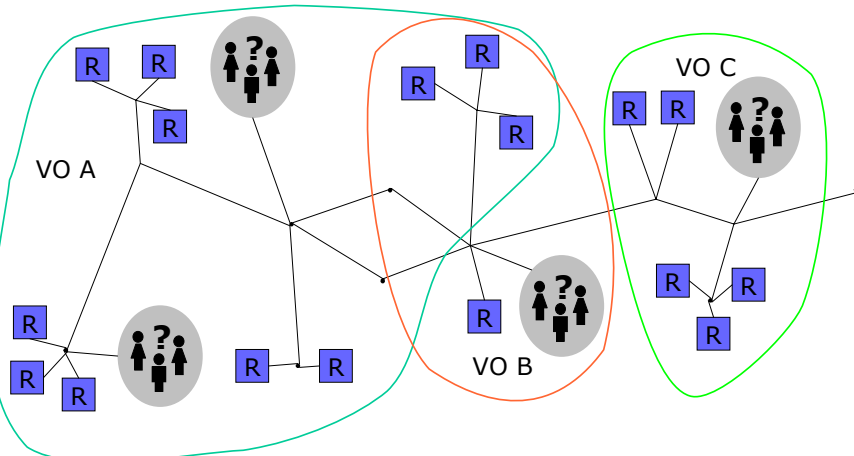


Grid Information Service

- Provide access to static and dynamic information regarding system components
- A basis for configuration and adaptation in heterogeneous, dynamic environments
- Requirements and characteristics
 - Uniform, flexible access to information
 - Scalable, efficient access to dynamic data
 - Access to multiple information sources
 - Decentralized maintenance



The GIS Problem: Many Information Sources, Many Views



Two Classes Of Information Servers

- **Resource Description Services**
 - Supplies information about a specific resource (e.g. Globus 1.1.3 GRIS).
- **Aggregate Directory Services**
 - Supplies collection of information which was gathered from multiple GRIS servers (e.g. Globus 1.1.3 GIIS).
 - Customized naming and indexing



Metacomputing Directory Service

- Use LDAP as Inquiry
- Access information in a distributed directory
 - Directory represented by collection of LDAP servers
 - Each server optimized for particular function
- Directory can be updated by:
 - Information providers and tools
 - Applications (i.e., users)
 - Backend tools which generate info on demand
- Information dynamically available to tools and applications



Two Classes Of MDS Servers

- Grid Resource Information Service (GRIS)
 - Supplies information about a specific resource
 - Configurable to support multiple information providers
 - LDAP as inquiry protocol
- Grid Index Information Service (GIIS)
 - Supplies collection of information which was gathered from multiple GRIS servers
 - Supports efficient queries against information which is spread across multiple GRIS server
 - LDAP as inquiry protocol



MDS Components

- LDAP 3.0 Protocol Engine
 - Based on OpenLDAP with custom backend
 - Integrated caching
- Information providers
 - Delivers resource information to backend
- APIs for accessing & updating MDS contents
 - C, Java, PERL (LDAP API, JNDI)
- Various tools for manipulating MDS contents
 - Command line tools, Shell scripts & GUIs



Grid Resource Information Service

- Server which runs on each resource
 - Given the resource DNS name, you can find the GRIS server (well known port = 2135)
- Provides resource specific information
 - Much of this information may be dynamic
 - > Load, process information, storage information, etc.
 - > GRIS gathers this information on demand
- “White pages” lookup of resource information
 - Ex: How much memory does machine have?
- “Yellow pages” lookup of resource options
 - Ex: Which queues on machine allows large jobs?

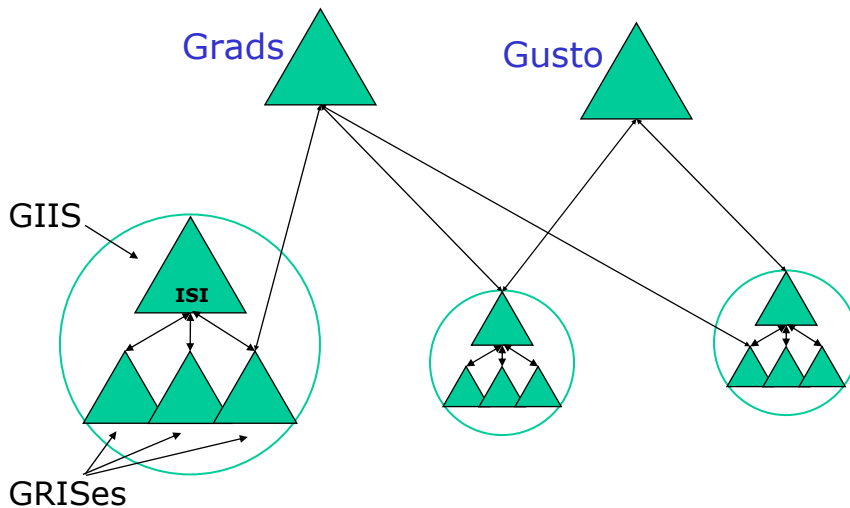


Grid Index Information Service

- **GIIS describes a class of servers**
 - Gathers information from multiple GRIS servers
 - Each GIIS is optimized for particular queries
 - > Ex1: Which Alliance machines are >16 process SGIs?
 - > Ex2: Which Alliance storage servers have >100Mbps bandwidth to host X?
 - Akin to web search engines
- **Organization GIIS**
 - The Globus Toolkit ships with one GIIS
 - Caches GRIS info with long update frequency
 - > Useful for queries across an organization that rely on relatively static information (Ex1 above)
- **Can be merged into GRIS**



Logical MDS Deployment



Resource Specifications

Resource Specifications

- Basic
 - » Globus RSL (Resource Specification Language)
 - » Condor Class Ads
- Constraint-based
 - » Redline
- Other:
 - » Sword (w/ value functions)
 - » vgDL (high level abstractions)
- Procedural Knowledge
 - » Recipe Based
 - » AI Planning

Globus RSL

```
specification := request
request := multirequest | conjunction | disjunction | parameter
multirequest := + request-list
conjunction := & request-list
disjunction := | request-list
request-list := ( request ) request-list | ( request )
parameter := parameter-name op value
op := = | > | < | >= | <= | !=
value := ([a..Z][0..9][ ])+
```

- A range of parameter names (literals which must be agreed)
- Simple Constraints on values
- Logical combinations of them

CSE225 – Lecture #4

RSL Examples

- A 100-node cluster, with 4GB nodes, with Oracle10g
 - » & (count = 100) (memory >=4GB) (database=Oracle10g)
- 4 32-node clusters, with 1, 2, 4, 1GB nodes, with Matlab
 - » & (& (count =32) (memory = 2GB) (database=Matlab))
 - » (& (count =32) (memory = 4GB) (database=Matlab))
 - » (& (count =32) (memory = 1GB) (database=Matlab))
 - » (& (count =32) (memory = 1GB) (database=Matlab))
- 2 128-node clusters, connected by a 1Gbit network
 - » & (& (count = 128) (network=1Gbit))
 - » (& (count=128) (network=1Gbit))

CSE225 – Lecture #4

Matchmaking

- ClassAds and suitable resources
 - » Allowable use: cpu, memory, load factor, allowable user
 - » Job requirements: cpu, memory, load factor, etc.
- Constraint $\leftarrow \rightarrow$ Compatibility
- Rank $\leftarrow \rightarrow$ Goodness Metric
- Matchmaker accepts Ads then introduces “matches”
- \Rightarrow selection = introduction
- \Rightarrow binding achieved by pairwise contact

CSE225 – Lecture #4

What Are ClassAds?

- A ClassAd maps attributes to expressions
- Expressions
 - » Constants: strings, numbers, etc.
 - » Expressions: `other.Memory > 600M`
 - » Lists: { “roy”, “pfc”, “melski” }
 - » Other ClassAds
- Powerful tool for grid computing
 - » Semi-structured (you pick your structure)
 - » Matchmaking

CSE225 – Lecture #4

ClassAd Example

```
[
  Type = "Job";
  Owner = "roy";
  Universe = "Standard";
  Requirements = (other.OpSys == "Linux" && other.DiskSpace >
  140M);
  Rank = (other.DiskSpace > 300M ? 10 : 1);
  ClusterID = 12314;
  JobID = 0;
  Env = "";
  ...
]
```

CSE225 – Lecture #4

ClassAd Matchmaking

```
[
  Type = "Job";
  Owner = "roy";
  Requirements = (other.OpSys == "Linux" && other.DiskSpace >
  140M);
  Rank = (other.DiskSpace > 300M ? 10 : 1);
]
[
  Type = "Machine";
  OpSys = "Linux";
  DiskSpace = 500M;
  AllowedUsers = {"roy", "melski", "pfc"};
  Requirements = (IsMember(other.Owner, AllowedUsers);
```

] CSE225 – Lecture #4

Matchmaking Extensions

- GangMatching
 - » Allows a class ad to include list of “ports” (each really a class ad with some scoping for shared names)
 - » A range of indexing and optimization techniques for implementation
 - » Matchmaker generates a “gang match” introduction
- Set Matching
 - » Allows expressions over aggregates (min, max, sum)
 - » Returns a set of class ads which meet the desired aggregate properties (and individual properties)

CSE225 – Lecture #4

A Constraint-based Approach

- Redline Language
 - » Define resource request as a set unbound variables (resource names) and constraints on them
 - » Define resource attributes as a set of constraints
- Match = binding of the resource names to specific resources
- Leverage wealth of investment and knowledge about constraint languages and solvers

CSE225 – Lecture #4

Redline Constraint Language

- Types=(real,int,string,boolean,UNDEF,ERR)
- Description= constraints*
- Constraints= variable op expression
 - » Expression = variable op variable | pred expression | value
- Predicates: minimize, maximize, forall X in SET, forany X in SET, required
- Sets, Lists, Enumerates
- Set Operations: count, max, min, sum, inSet, set_intersect, set_union, set_difference, set_s_difference
- => Full-blown constraint language, with range of intermediate variables and terms

CSE225 – Lecture #4

Redline Examples

```
[user="globus-user";
Group="dsl-uc";
Computation ISA
  SET[type="computation"];
Storage ISA [type="storage";
  space>100];
Forall x in computation;
x.cpuspeed>150;
x.bandwidth[storage.hn]>30;
x.accesstime>80;
Sum(computation.memory)>300;
Storage.space>80;
Storage.accesstime>18]
```

```
R1=[type="computation";
  hn="ucsd1";cpuspeed=200;
  bandwidth=DICT[{"s1",20},{"s2",40}];
  accesstime>17]
R2=[type="computation";
  hn="ucsd2";cpuspeed=200;
  bandwidth=DICT[{"s1",20},{"s2",40}];
  accesstime>17]
R3=[type="storage";hn="s1";space=100]
R4=[type="storage";hn="s2";space=200]
```

CSE225 – Lecture #4

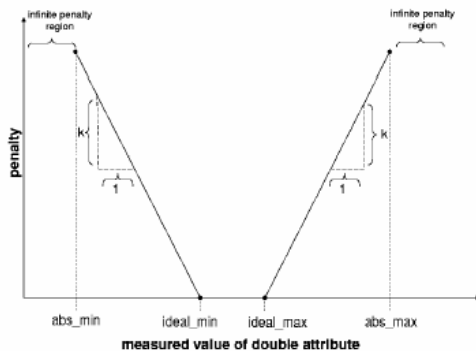
Benefits

- Tremendous Leverage
 - » Well understood language and solver properties
 - » Well developed implementations and solvers
- Regular framework is easily extensible
- Full constraint language is VERY powerful
 - » All of the things in other languages
 - » General queries, incremental changes
 - » Full symmetry and enumeration

CSE225 – Lecture #4

SWORD

```
<request>
<dist_query_budget>30</dist_query_budget>
<optimizer_budget>70</optimizer_budget>
<group>
<name>Cluster_NA</name>
<num_machines>4</num_machines>
<cpu_load>0.0, 0.0, 1.0, 2.0, 0.01</cpu_load>
<free_mem>256.0, 512.0, MAX, MAX, 1.0</free_mem>
<free_disk>500.0, 10000.0, MAX, MAX, 5.0</free_disk>
<latency>0.0, 0.0, 10.0, 20.0, 0.5</latency>
<os>
<value>Linux, 0.0</value>
</os>
<network_coordinate_center>
<value>North_America, 0.0</value>
</network_coordinate_center>
</group>
<group>
<name>Cluster_Europe</name>
<num_machines>4</num_machines>
<cpu_load>0.0, 0.0, 1.0, 2.0, 0.01</cpu_load>
<free_mem>256.0, 512.0, MAX, MAX, 1.0</free_mem>
<free_disk>100.0, 10000.0, MAX, MAX, 5.0</free_disk>
<latency>0.0, 0.0, 10.0, 20.0, 0.5</latency>
<os>
<value>Linux, 0.0</value>
</os>
<network_coordinate_center>
<value>Europe, 0.0</value>
</network_coordinate_center>
</group>
<constraint>
<group_names>Cluster_NA Cluster_Europe</group_names>
<latency>0.0,0.0,0.0,50.0,100.0, 0.5</latency>
</constraint>
</request>
```



Virtual Grid Description Language (vgDL)

- Sophisticated grid application developers use a high-level resource abstractions to organize their applications
- vgDL provides convenient high level descriptions
 - » Aggregators: Cluster, TightBag, LooseBag
 - » Couplers: HighBW, LowBW, Near
 - » =>Responses tuned with Ranking Function
- See <http://www-csag.ucsd.edu/projects/VGrADS/>

CSE225 – Lecture #4

vgDL Example

- BigBagOfClusters3 = LooseBagOf(N)[1:1000000]
- [Rank = LooseBag.Memory + LooseBag.Nodes]
- {
- N = ClusterOf(M)[8:32]
- {
- M = [(Memory >= 1024)&&(Disk > 2048)]
- [Rank = Clock]
- }
- }

CSE225 – Lecture #4

Model for all of these...

- Globus RSL (Resource Specification Language)
- Condor Class Ads
- Redline
- Sword (w/ value functions)
- vgDL (high level abstractions)

Model of Use

1. Application supplies declarative specification
2. System Selects one or more Candidate Sets
3. Returns to Application
4. Application Goes to Allocate/Bind Resources

CSE225 – Lecture #4

A Second Type of Description

- Other types of Knowledge may be helpful
- Complex relationships hard to express
- Heuristics/strategies for realizing good configurations
 - » Design Approaches/Principles
- Other Unusual Constraints/Properties hard to express
- Two Schemes
 - » Domain Expert Knowledge
 - » General AI Planning Knowledge

CSE225 – Lecture #4

Recipe-Based Service Composition (Cheng&Huang)

- Service Recipes
 - » Operational Description of Service Specific Knowledge
 - » Abstract Mapping Knowledge
 - Composition of Service Components into Graph-like Structure
 - Functional Requirements
 - » Physical Mapping Knowledge
 - Realization of the Abstract Map on an Actual Physical Configuration
 - Define Objectives which drive Heuristics (what's important!)
- Accept a Series of User Requests
 - » Implement Identical High Level
 - » Parametric, other similar variants

CSE225 – Lecture #4

Example

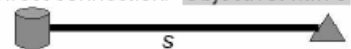
- Abstract Mapping (Unshaded) and Physical Mapping (Shaded)

1 Add MPEG-2 server



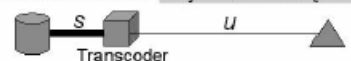
2 If client wants MPEG-2:

Direct connection. Objective: $\min s$



3 Otherwise:

Use transcoder. Objective: $\min (R \cdot s + u)$



```

AbsConf conf = new AbsConf();
Term obj = new Term(new Double(0.0));
AbsComp vserver = conf.addComp("MPEG2VideoServer");

if (client.getProperty("VideoIn").equals("MPEG2")) {
  conf.addConn(vserver, client);
  obj.add(new Term(new LatencyM(vserver, client)));
} else {
  AbsComp transcoder = conf.addComp("Transcoder");
  conf.addConn(vserver, transcoder);
  Term partialObj = new Term(new LatencyM(vserver, transcoder));
  partialObj.multiplyBy(new Term(new Double(MPEG2BitRate)));
  obj.add(partialObj);
  conf.addConn(transcoder, client);
  partialObj = new Term(new LatencyM(transcoder, client));
  partialObj.multiplyBy(new Term(new Double(MPEG4BitRate)));
  obj.add(partialObj);
}
Function objfunc = new Function(obj);
  
```

CSE225 – Lecture #4

Sekitei (Kichkaylo & Karamcheti)

- Use a Generic Planner
 - » Component Specifications
 - » Ability to Express and Manage all kinds of Constraints and Limitations (Plans around them)
 - » Integrates notion of binding and constraints that cannot be easily expresses
 - » Real Resource Environments
- ...definitely read this paper...

CSE225 – Lecture #4

Discussion

- Expression of Resource Needs
- Complexity of Resource Descriptions
- Can a developer actually get this right?
- Dynamic Performance Information
- Selection vs. Binding

CSE225 – Lecture #4

Summary

- Wide Range of Resource Description Languages
- Many try to describe declaratively the needed resources precisely (RSL, Class Ads, Redline, SWORD)
- A New one, Describes Collections and Organization, but leaves details vague (vgDL)
- Two combine imperative approaches to handle more complex resource environments and exploit service knowledge (Recipes, Sekitei)
- Next time: Implementing Selection... (Nut Taesombut Guest Lecture)