Parallel & Distributed Computing (WQD7008)

Week 6

Grid Computing and Management of Distributed Systems

2019/2020 Semester 1

Dr. Hamid Tahaei

Grid Architecture and Service Modeling

GRID ARCHITECTURE AND SERVICE MODELING

- What is Metacomputing
  - a technology designed to integrate multiple computing resources to develop a variety of applications related to business, management, industry and software. Metacomputing technology is also used to gather, interpret and analyze data from various databases and devices.
- Grid
  - is a metacomputing infrastructure that brings together computers (PCs, workstations, server clusters, supercomputers, laptops, notebooks, mobile computers, PDAs, etc.) to form a large collection of compute, storage, and network resources.
  - to solve large-scale computation problems or to enable fast information retrieval.
  - The coupling between hardware and software with special user applications is achieved by leasing the hardware, software, middleware, databases, instruments, and networks as computing utilities. Good examples include the renting of expensive special-purpose application software on demand and transparent access to human genome databases.

Grid Architecture and Service Modeling

GRID ARCHITECTURE AND SERVICE MODELING

- **Goal:**
  - To explore fast solutions for large-scale computing problems.
    - This objective is shared by computer clusters and massively parallel processor (MPP) systems.
- Grid computing takes advantage of the existing computing resources scattered in a nation or internationally around the globe. In grids, resources owned by different organizations are aggregated together and shared by many users in collective applications.
- Grids rely heavy use of LAN/WAN resources across enterprises, organizations, and governments. The virtual organizations or virtual supercomputers are new concept derived from grid or cloud computing. These are virtual resources dynamically configured and are not under the full control of any single user or local administrator.

Grid Architecture and Service Modeling

~~**Grid History and Service Families:**~~

- The Internet was developed in the 1980s to provide computer-to-computer connections.
  - telnet:// protocol
- The web service was developed in the 1990s to establish direct linkage between web pages.
  - http:// protocol
- From 1990, grids became gradually available to establish large pools of shared resources.
  - The approach is to link many Internet applications across machine platforms directly in order to eliminate isolated resource islands.
- Grids and clusters are not the same:
  - Cluster nodes are more homogeneous machines that are better coordinated to work collectively and cooperatively. The grid nodes are heterogeneous computers that are more loosely coupled together over geographically dispersed sites.

Grid Architecture and Service Modeling

~~**Grid History and Service Families:**~~

- Fathers of the grids:
  - The idea of the grid was pioneered by Ian Foster, Carl Kesselman and Steve Tuecke in a 2001.

- Four major families of grid computing systems were suggested by the Forbes GGG om 2001

Table 7.1 Four Grid Families Identified in the Great Global Grid (GGG) (*Forbes*, 2001)

Grid Family	Representative Grid Systems and References
Computational Grids or Data Grids	TeraGrid (US), EGEE (EU), DataGrid (EU), Grid'5000 (france), ChinaGrid (China), NAS (NASA), LCG (Cern), e-Science (UK), D-Grid (Nordic), FutureGrid (US), etc.
Information Grids or Knowledge Grids	Semantic Grid, Ontology Platform, BOINC (Berkeley), D4Science, Einsten@Home, Information Power Grid (NASA)
Business Grids	BEinGrid (EU), HP eSpeak, IBM WebSphere, Sun Grid Engine, Microsoft .NET, etc.
P2P/Volunteer Grids	SETI@Home, Parasic Grid, FightAIDS@Home, Foldong@Home, GIMPS, etc.

Grid Architecture and Service Modeling

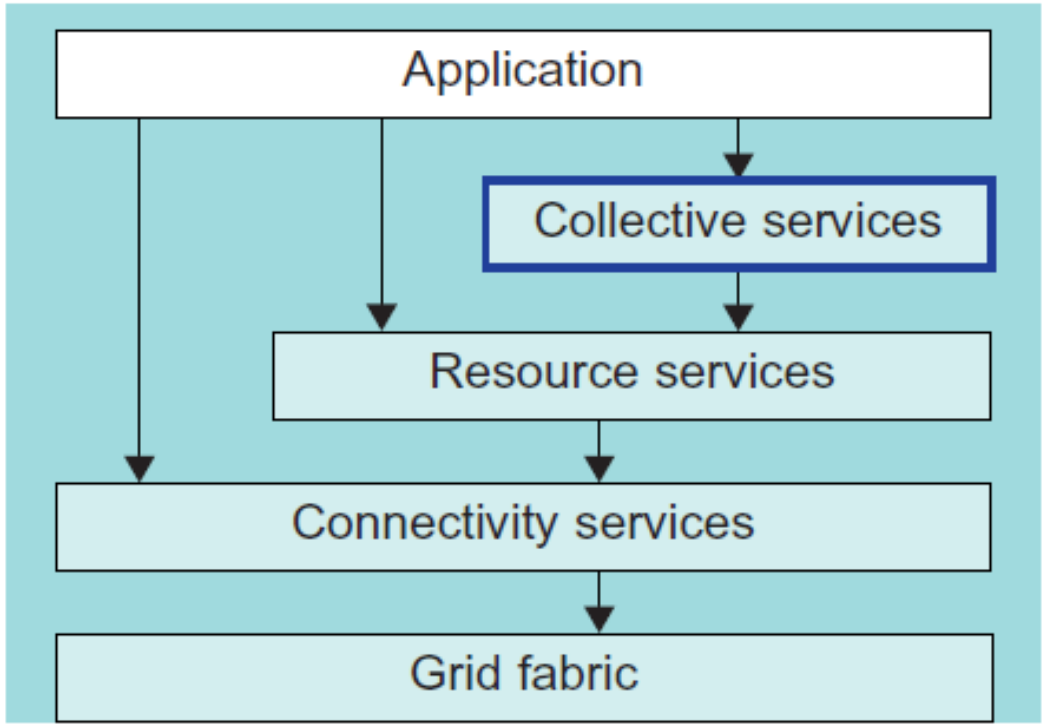
~~**Grid History and Service Families:**~~

Four Grid Service Families:

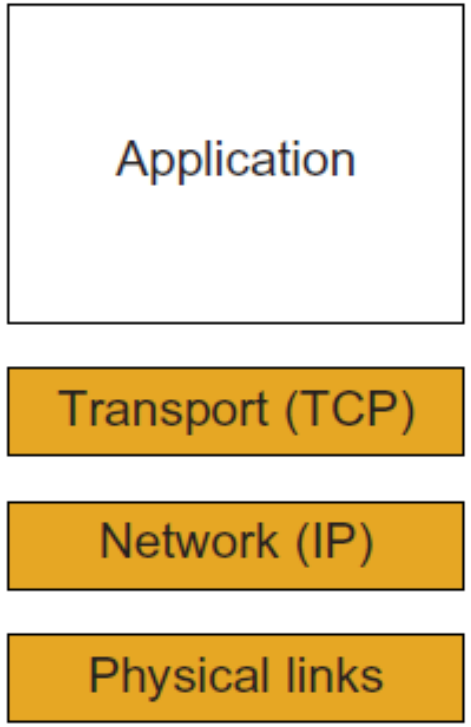
- Computational grids or data grids:
  - Most of today's grid systems.
  - Good examples are the NSF TeraGrid installed in the United States and the DataGrid built in the European Union.
- Information or knowledge grids:
  - Dedicated to knowledge management and distributed ontology processing. The Semantic web, also known as semantic grids, belongs to this family.
  - Ontology platform falls into information or knowledge grids. Other information/knowledge grids include the Berkeley BOINC and NASA's Information Power Grid.
- Business grids:
  - built for business data/information processing. These are represented by the HP eSpeak, IBM WebSphere, Microsoft .NET, and Sun One systems.
  - Some business grids are being transformed into Internet clouds.
- P2P grids and parasitic grids:
  - Can use grid or P2P technologies for organization and management of services.

Grid Architecture and Service Modeling

Grid Service Protocol Stack:



Grid service protocol stack



Internet protocol

Grid Architecture and Service Modeling

Grid Resources:

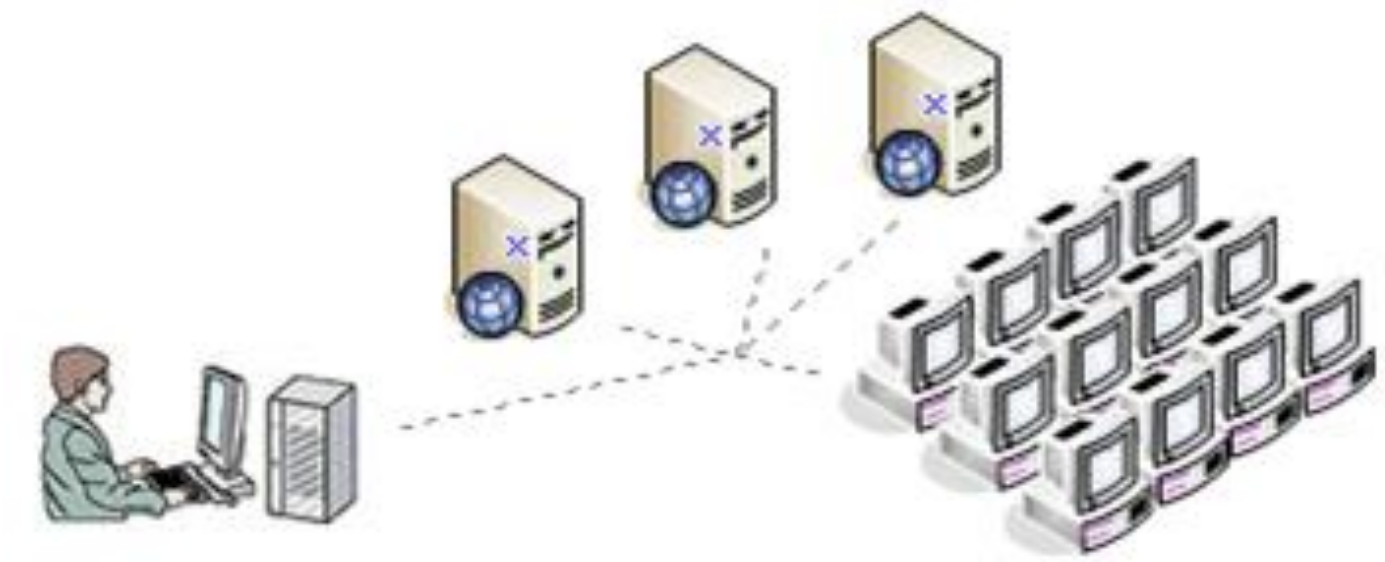
Table 7.2 Control Operations and Enquiries for Aggregating Grid Resources

Resources	Control Operations	Enquiries
Compute resources	Starting, monitoring, and controlling the execution of resultant processes; control over resources: advance reservation	Hardware and software characteristics; relevant load information: current load and queue state
Storage resources	Putting and getting files; control over resources allocated to data transfers: advance reservation	Hardware and software characteristics; relevant load information: available space and bandwidth utilization
Network resources	Control over resources allocated	Network characteristics and load
Code repositories	Managing versioned source and object code	Software files and compile support
Service catalogs	Implementing catalog query and update operations: a relational database	Service order information and agreements

Grid Architecture and Service Modeling

CPU Scavenging and Virtual Supercomputers:

- The concept of creating a “grid” from the unused resources in a network of computers is known as CPU scavenging.
- In reality, virtual grids are built over large number of desktop computers by using their free cycles at night or during inactive usage periods. The donors are ordinary citizens on a voluntary participation basis.
- In practice, these client hosts also donate some disk space, RAM, and network bandwidth in addition to the raw CPU cycles. At present, many volunteer computing grids are built using the CPU scavenging model.



Grid Architecture and Service Modeling

CPU Scavenging and Virtual Supercomputers:

- Both public and virtual grids can be built over large or small machines, that are loosely coupled together to satisfy the application need.
- Grid vs the conventional supercomputers, in the context of distributed computing :
  - Supercomputers like MPPs in the Top-500 list are more homogeneously structured with tightly coupled operations
  - Grids are built with heterogeneous nodes running non-interactive workloads. These grid workloads may involve a large number of files and individual users.
  - The geographically dispersed grids are more scalable and fault-tolerant with significantly lower operational costs than the supercomputers.

Grid Architecture and Service Modeling

CPU Scavenging and Virtual Supercomputers - Grid Resource Aggregation

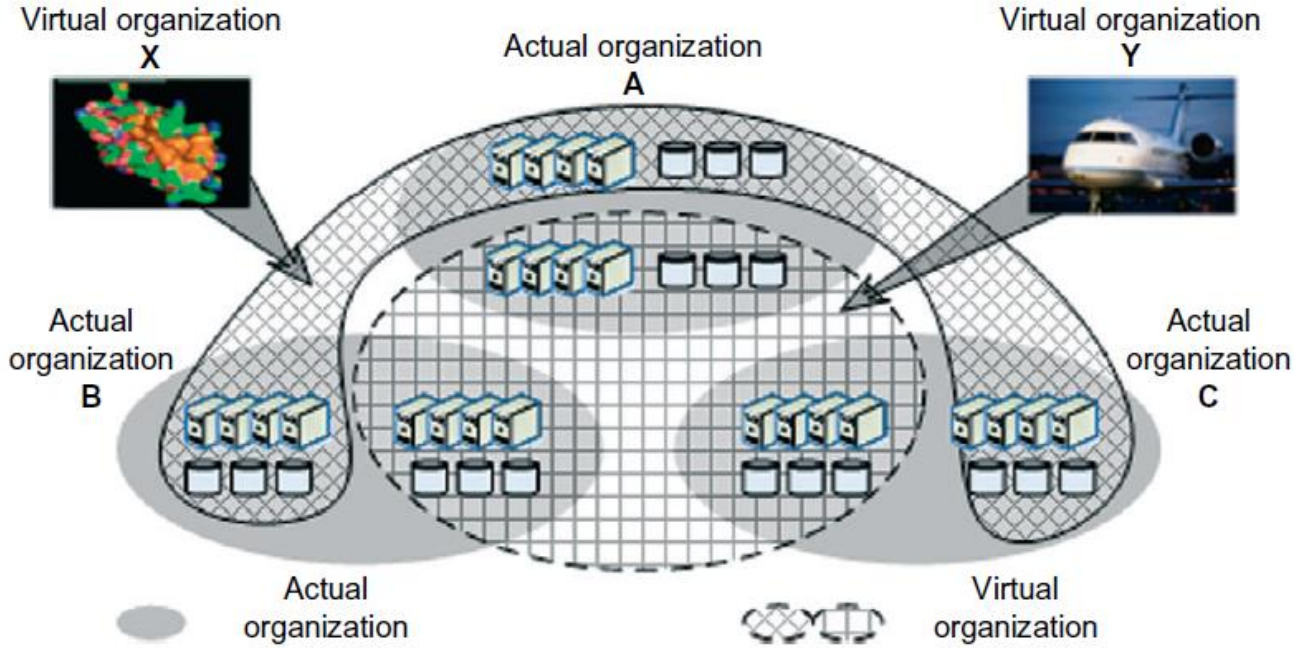
Several assumptions are made for grids or clouds during the resource aggregation process:

- The compute nodes and other necessary resources for grids do not join or leave the system incidentally, except when some serious faults occur in the grid.
- cloud resources are mostly provisioned from large data centers. Since security and reliability are very tight in these data centers, resource behaviour is not predictable.
- although resources in P2P systems are casually allocated, we can build P2P grids for distributed file sharing, content delivery, gaming, and entertainment applications. The joining or leaving of some peers has little impact on the needed functions of a P2P grid system.
- Today's grid computing applications are no longer restricted to using HPC systems. HTC systems, like clouds, are even more in demand in business services.

Grid Architecture and Service Modeling

CPU Scavenging and Virtual Supercomputers - Virtual Organization:

- The grid is a distributed system integrated from shared resources to form a virtual organization (VO). The VO offers dynamic cooperation built over multiple physical organizations. The virtual resources contributed by these real organizations are managed autonomously. The grid must deal with the trust relationship in a VO. The applications in a grid vary in terms of workload and resource demand. A flexible grid system should be designed to adapt to varying workloads. In reality, physical organizations include a real company, a university, or a branch of government. These real organizations often share some common objectives.



Grid Architecture and Service Modeling

Open Grid Services Architecture (OGSA):

The OGSA is extended from web service concepts and technologies. The standard defines a common framework that allows businesses to build grid platforms across enterprises and business partners.

- Compared with the layered grid architecture, the OGSA is service-oriented. A service is an entity that provides some capability to its client by exchanging messages (Week 5).
- The service-oriented architecture (SOA) serves as the foundation of grid computing services. The individual and collective states of resources are specified in this service standard. The standard also specifies interactions between these services within the particular SOA for grids. An important point is that the architecture is not layered, where the implementation of one service is built upon modules that are logically dependent. One may classify this framework as object-oriented. Many web service standards, semantics, and extensions are applied or modified in the OGSA.

Grid Architecture and Service Modeling

Open Grid Services Architecture - OGSA Interfaces:

Table 7.3 OGSA Grid Service Interfaces Developed by the OGSA Working Group

Port Type	Operation	Brief Description
Grid service	Find service data	Query a grid service instance, including the handle, reference, primary key, home handle map, interface information, and service-specific information. Extensible support for various query languages.
	Termination time	Set (and get) termination time for grid service instance.
	Destroy	Terminate grid service instance.
Notification source	Subscribe to notification topic	Subscribe to notifications of service events. Allow delivery via third-party messaging services.
Notification sink	Deliver notification	Carry out asynchronous delivery of notification messages.
Registry	Register service	Conduct soft-state registration of Grid Service Handles (GSHs).
	Unregister service	Unregister a GSH.
Factory	Create service	Create a new grid service instance.
Handle map	Find by handle	Return the Grid Service Reference (GSR) associated with the GSH.

Grid Architecture and Service Modeling

Open Grid Services Architecture - Grid Service Handle (GSH):

- A GSH is a globally unique name that distinguishes a specific grid service instance from all others. The status of a grid service instance could be that it exists now or that it will exist in the future. These instances carry no protocol or instance-specific addresses or supported protocol bindings. Instead, these information items are encapsulated along with all other instance-specific information. In order to interact with a specific service instance, a single abstraction is defined as a GSR.
- Unlike a GSH, which is time-invariant, the GSR for an instance can change over the lifetime of the service. The OGSA employs a “handle-resolution” mechanism for mapping from a GSH to a GSR. The GSH must be globally defined for a particular instance. However, the GSH may not always refer to the same network address. A service instance may be implemented in its own way, as long as it obeys the associated semantics.

Grid Architecture and Service Modeling

Grid Service Migration:

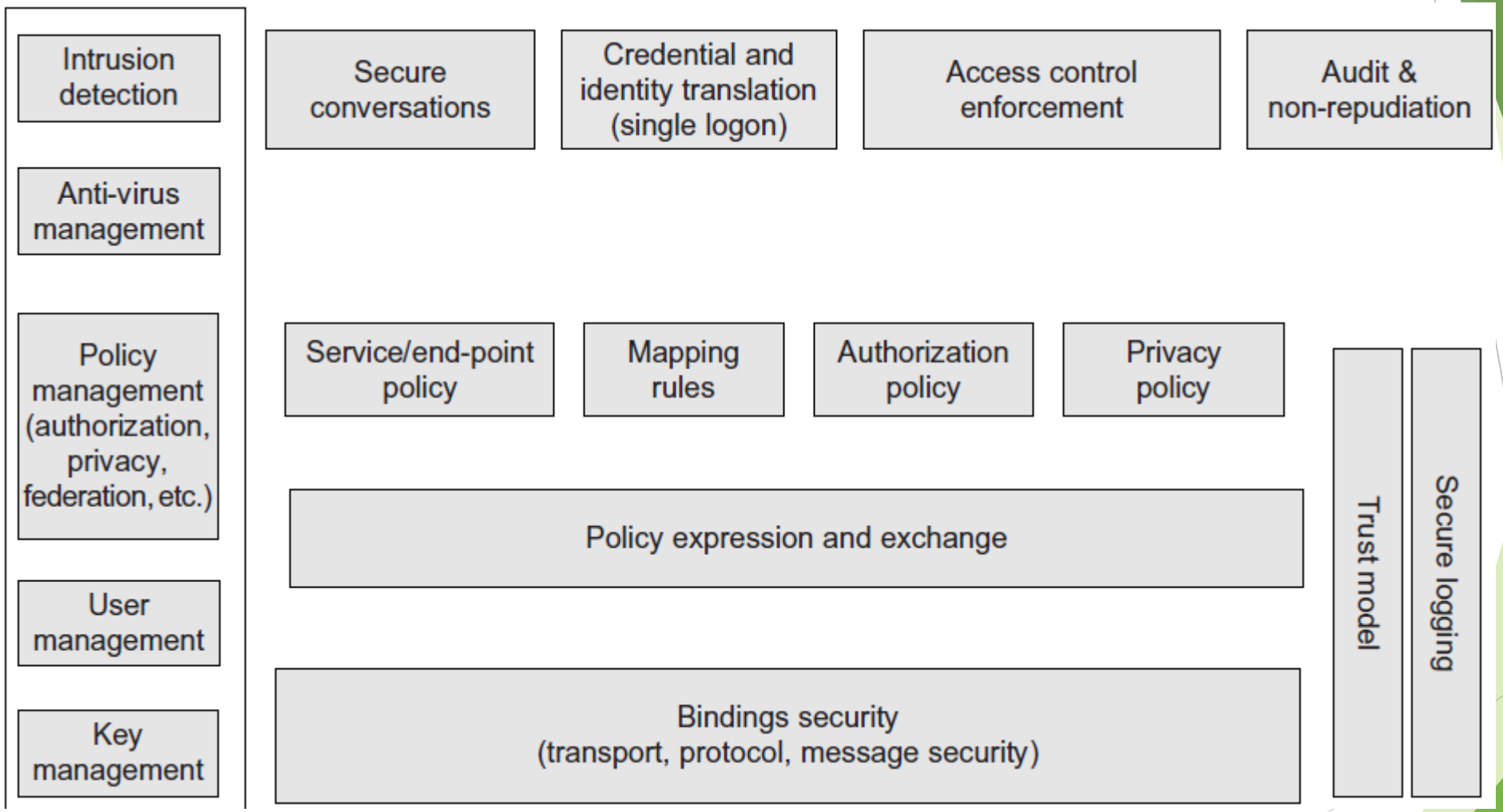
- This is a mechanism for creating new services and specifying assertions regarding the lifetime of a service. The OGSA model defines a standard interface, known as a factor, to implement this reference. Any service that is created must address the former services as the reference of later services. This creates a requested grid service with a specified interface and returns the GSH and initial GSR for the new service instance. It should also register the new service instance with a handle resolution service. Each dynamically created grid service instance is associated with a specified lifetime.

Grid Architecture and Service Modeling

OGSA Security Models:

- The grid works in a heterogeneous distributed environment, which is essentially open to the general public. It must be able to detect intrusions or stop viruses from spreading by implementing secure conversations, single logon, access control, and auditing for nonrepudiation. At the security policy and user levels, we want to apply a service or endpoint policy, resource mapping rules, authorized access of critical resources, and privacy protection. At the Public Key Infrastructure (PKI) service level, the OGSA demands security binding with the security protocol stack and bridging of certificate authorities (CAs), use of multiple trusted intermediaries, and so on. Trust models and secure logging are often practiced in grid platforms.

Grid Architecture and Service Modeling



Grid Architecture and Service Modeling

Data-Intensive Grid Service Models:

- Applications in the grid are normally grouped into two categories: computation-intensive and data intensive. For data-intensive applications, we may have to deal with massive amounts of data. For example, the data produced annually by a Large Hadron Collider may exceed several petabytes (10^15 bytes). The grid system must be specially designed to discover, transfer, and manipulate these massive data sets. Transferring massive data sets is a time-consuming task. Efficient data management demands low-cost storage and high-speed data movement.

Grid Architecture and Service Modeling

Data-Intensive Grid Service Models:

- Several common methods for solving data movement problems

Data Replication and Unified Namespace

By replicating the same data blocks and scattering them in multiple regions

Grid Data Access Models

There are four access models for organizing a data grid

- Monadic model
- Hierarchical model
- Federation model
- Hybrid model

Parallel versus Striped Data Transfers

Parallel data transfer opens multiple data streams for passing subdivided segments of a file simultaneously

Grid Architecture and Service Modeling

Data Replication and Unified Namespace

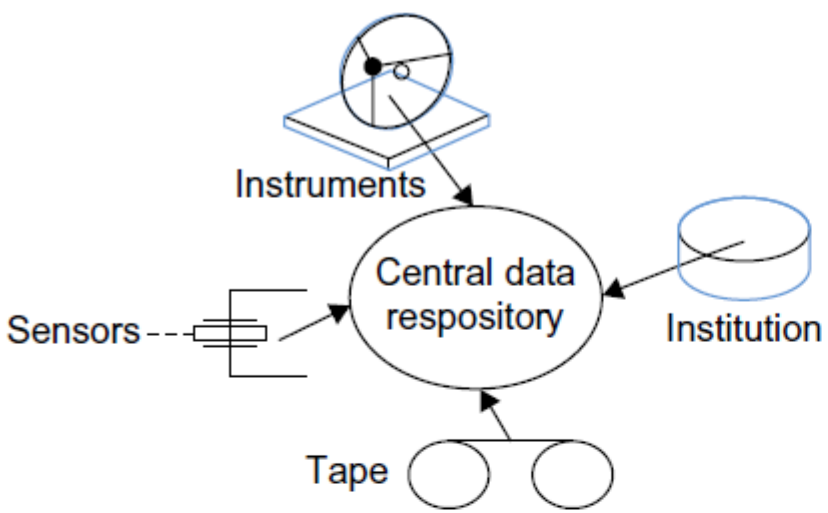
- This data access method is also known as caching, which is often applied to enhance data efficiency in a grid environment. By replicating the same data blocks and scattering them in multiple regions of a grid, users can access the same data with locality of references. Furthermore, the replicas of the same data set can be a backup for one another. Some key data will not be lost in case of failures. However, data replication may demand periodic consistency checks. The increase in storage requirements and network bandwidth may cause additional problems.
- Replication strategies determine when and where to create a replica of the data. The factors to consider include data demand, network conditions, and transfer cost. The strategies of replication can be classified into method types: dynamic and static. For the static method, the locations and number of replicas are determined in advance and will not be modified. Although replication operations require little overhead, static strategies cannot adapt to changes in demand, bandwidth, and storage vailability. Dynamic strategies can adjust locations and number of data replicas according to changes in conditions (e.g., user behavior).
- However, frequent data-moving operations can result in much more overhead than in static strategies. The replication strategy must be optimized with respect to the status of data replicas. For static replication, optimization is required to determine the location and number of data replicas. For dynamic replication, optimization may be determined based on whether the data replica is being created, deleted, or moved. The most common replication strategies include preserving locality, minimizing update costs, and maximizing profits.

Grid Architecture and Service Modeling

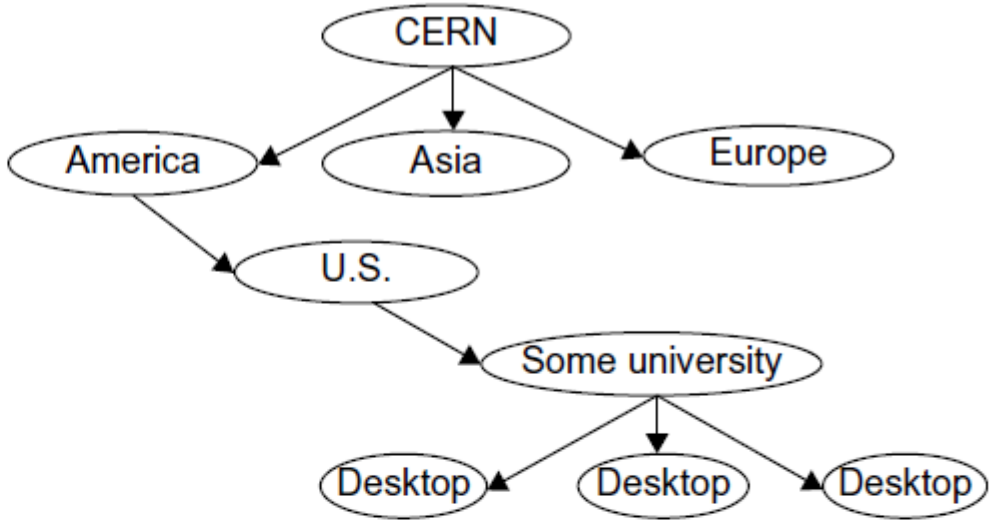
Grid Data Access Models

- Multiple participants may want to share the same data collection. To retrieve any piece of data, we need a grid with a unique global namespace. Similarly, we desire to have unique file names. To achieve these, we have to resolve inconsistencies among multiple data objects bearing the same name. Access restrictions may be imposed to avoid confusion. Also, data needs to be protected to avoid leakage and damage. Users who want to access data have to be authenticated first and then authorized for access.
- Monadic model
- Hierarchical model
- Federation model
- Hybrid model

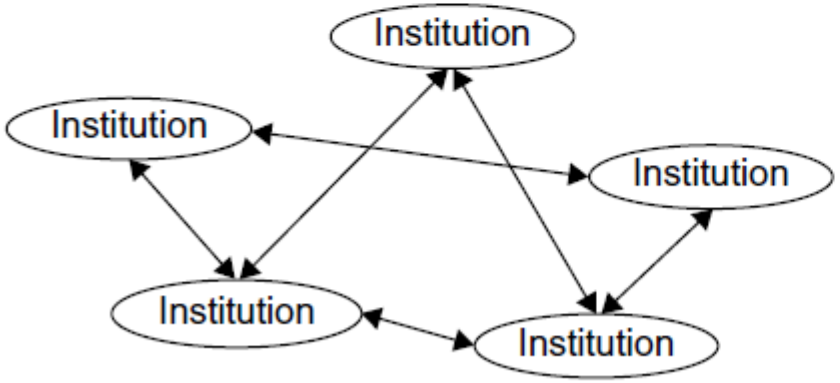
Grid Architecture and Service Modeling



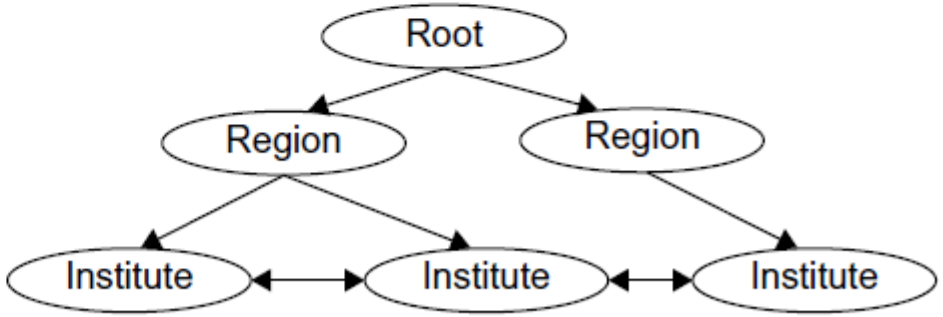
(a) Monadic



(b) Hierarchy



(c) Federation



(d) Hybrid

Grid Architecture and Service Modeling

Parallel versus Striped Data Transfers:

- Compared with traditional FTP data transfer, parallel data transfer opens multiple data streams for passing subdivided segments of a file simultaneously. Although the speed of each stream is the same as in sequential streaming, the total time to move data in all streams can be significantly reduced compared to FTP transfer. In striped data transfer, a data object is partitioned into a number of sections, and each section is placed in an individual site in a data grid. When a user requests this piece of data, a data stream is created for each site, and all the sections of data objects are transferred simultaneously. Striped data transfer can utilize the bandwidths of multiple sites more efficiently to speed up data transfer.

Grid Projects and Grid Systems Built

- **National or international**
- National grids are mainly funded through government sources. These national grids are developed to promote research discovery, middleware products, and utility computing in grid-enabled applications.
- Most national grids are built by linking supercomputer centers and major computer ensembles together with Internet backbones and high-bandwidth WANs or LANs.
- United States
- European Union
- United Kingdom
- France
- China.

Grid Projects and Grid Systems Built

- National Grid Computing

Table 7.4 Some National Grid Projects in Five Countries

Project	Sponsor, Launch	Distributed Computing Capability and Applications
TeraGrid (USA)	NSF, 2002 (Sec. 7.2.2)	An open grid infrastructure in 11 resource sites, with 40 Gbps Internet2 backbone links with 2 Pflops computing power, 50 PB storage, and 100 specific databases.
DataGrid (EU)	EU and CERN, 2001 (7.2.3)	Claimed the largest grid on earth for data analysis in high-energy physics, environment, and bioinformatics, etc.
Grid'5000 (France)	French Gov., 2006, (Sec. 7.2.1)	An experimental grid comprising 5,000 processor cores in 9 French cities for HPC and research use (Section 7.2.2).
ChinaGrid (China)	Education Ministry 2005, (Sec. 7.2.4)	An educational computing grid for research use, linking HPC systems in 100 Chinese universities.
NAS Grid (USA)	NASA Ames Lab, 2008	Running genetic algorithms with the Condor cycle scavenger on large number of Sun and SGI workstations.

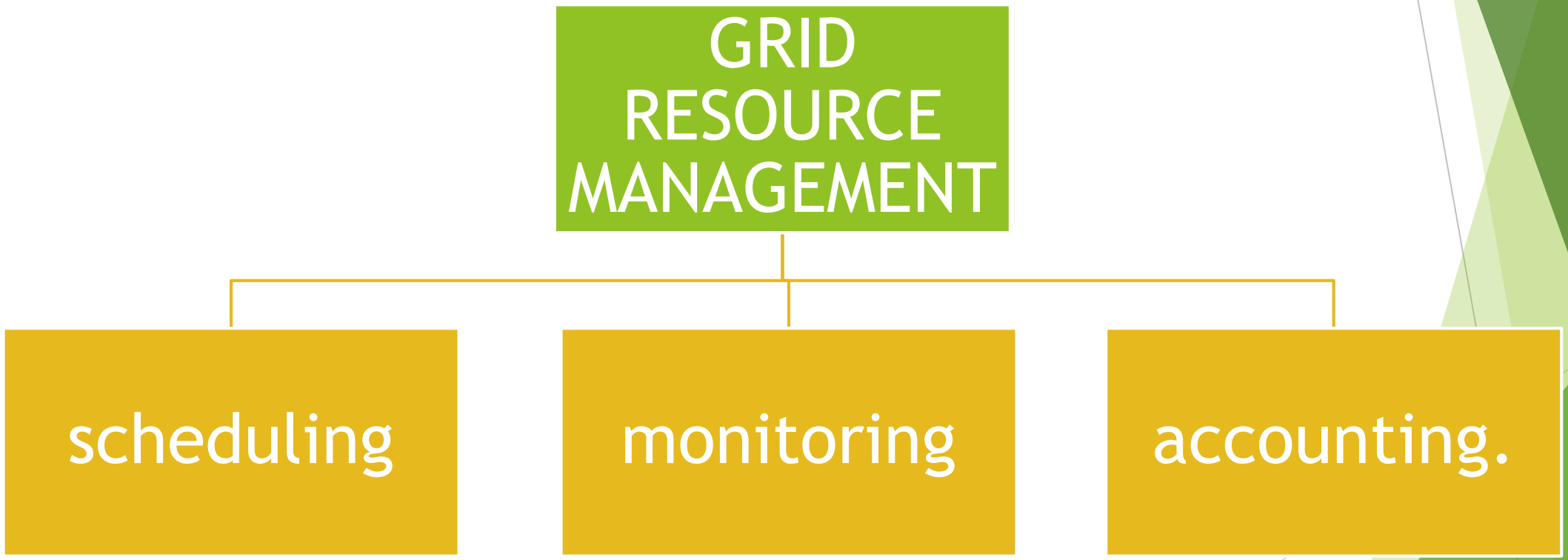
Grid Projects and Grid Systems Built

- International Grid

Table 7.5 International Grid Projects (Past and Current)

Project	Description and Operating Region	Status
EGEE	Enabling Grids for E-sciencE in European Union countries	2004–2010
D4Science	Grid-enabled technology for science in Europe, Asia and the Pacific	2008–2009
Nordic D-Grid	Nordic Data grid facility in Scandinavia and Finland	2006–2010
Future Grid	High-performance and grid research groups in U.S. universities	2010–present
WorldGrid	World community grid has 6.2 million machines running the BOINC in 2011	2004–present
SETI@Home	Volunteer grid with 2.5 machines running radioscope signals in 2001	2001–present
BOINC	Berkeley-led volunteer grid projects including the PrimeGrid, DNETC2Home, MilkyWay@Home, Collatz Conjective, etc.	2002–present
BEinGRID	Business Experiments in Grid funded by the European Commission	2006–2009

GRID RESOURCE MANAGEMENT AND BROKERING

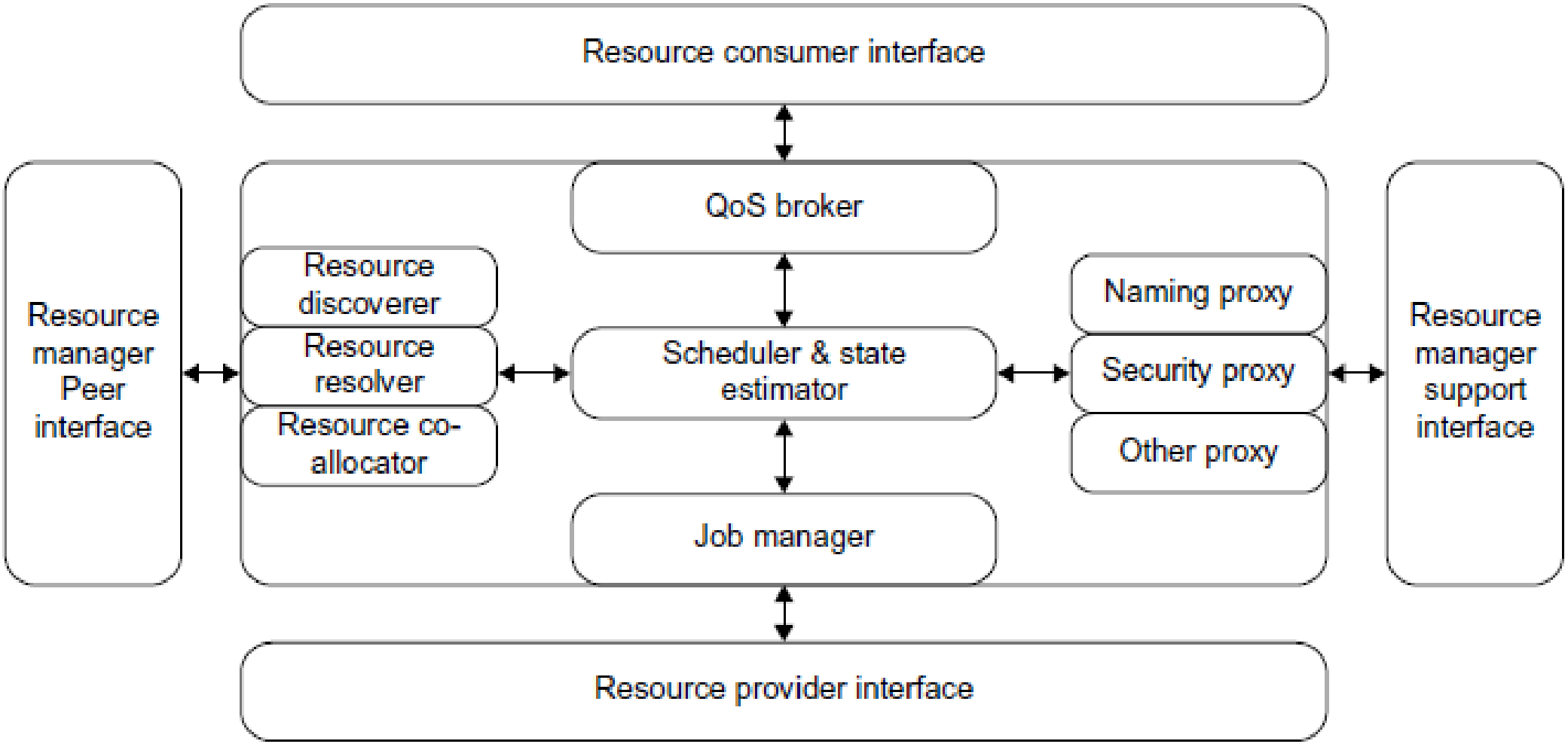


GRID RESOURCE MANAGEMENT AND BROKERING

Resource Management and Job Scheduling

- In a grid system, resources are usually autonomous. Each organization may have its own resource management policies.
- It is more reasonable and adaptable to set an individual resource management system (RMS) for an organization—the RMSes in the upper level can be considered the resource consumers, and the RMSes in the lower level can be considered the resource providers.
- To support such a multi-RMS structure, an abstract model for RMS is introduced.
- Several published interfaces in the model:
  - The resource consumer interface: is for access from upper-level RMS or user applications.
  - The resource discoverer: actively searches qualified resources by this interface.
  - The resource disseminator: broadcasts information about local resources to other RMSes.
  - The resource trader: exchanges resources among RMSes for market-based grid systems.
  - The resource resolver: routes jobs to remote RMSs.
  - The resource co-allocator simultaneously allocates multiple resources to a job.

GRID RESOURCE MANAGEMENT AND BROKERING



GRID RESOURCE MANAGEMENT AND BROKERING

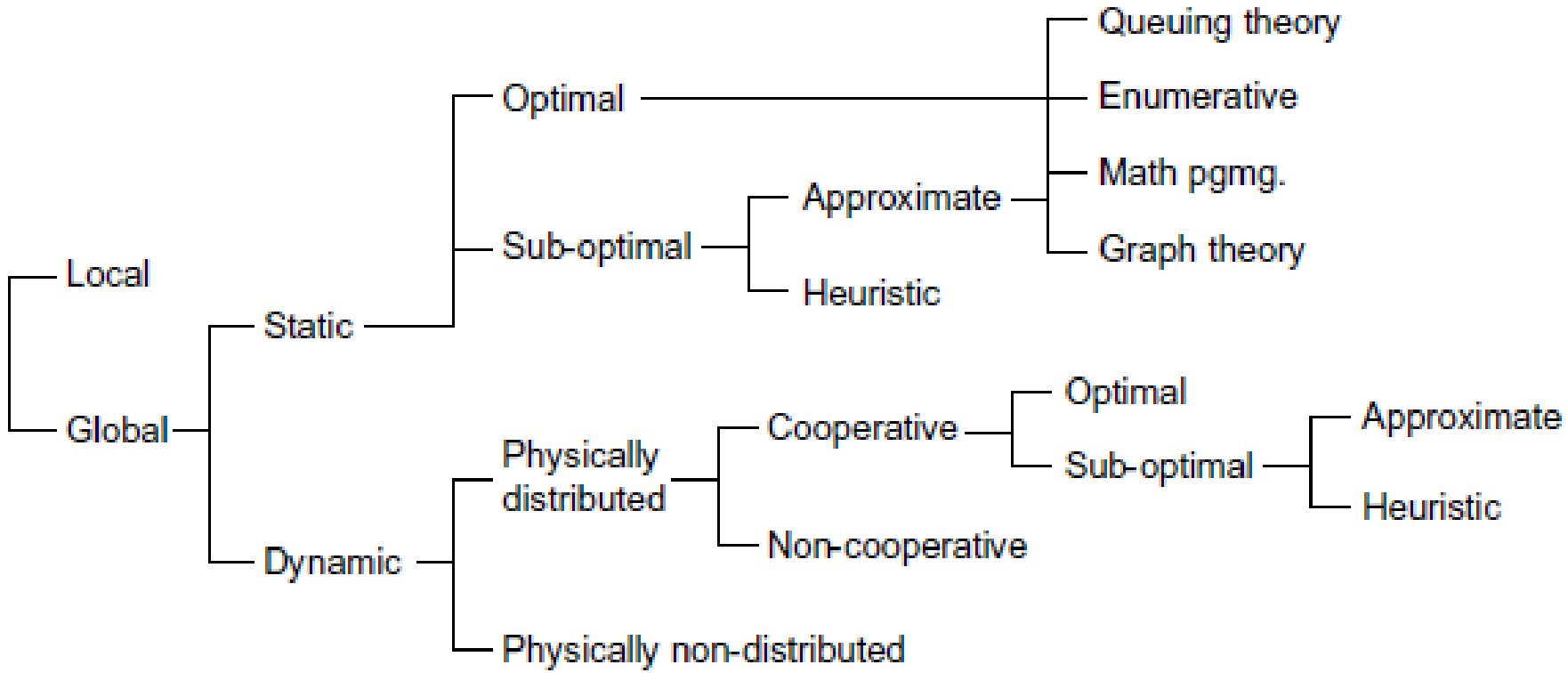
Hierarchy of RMSes

- RMSes are interconnected with one another. The RMSes are set in a hierarchical structure. The resource provider interface is for the lower-level RMS or actual resources. Resources and jobs in the lower-level RMS are managed and controlled through this interface. If it is supported, a resource reservation agent is also implemented here. A resource manager peer interface is used to interoperate with other RMSes in the same level. Job requests are accepted from this interface, and are forwarded to the local scheduler.
- Four published interfaces in the model:
- The resource consumer interface is for access from upper-level RMS or user applications.
- The resource discoverer actively searches qualified resources by this interface.
- The resource disseminator broadcasts information about local resources to other RMSes.
- The resource trader exchanges resources among RMSes for market-based grid systems.
- The resource resolver routes jobs to remote RMSes.
- The resource co-allocator simultaneously allocates multiple resources to a job.

GRID RESOURCE MANAGEMENT AND BROKERING

Grid Job Scheduling Methods

- Two schemas are often used to classify on scheduling technology:
  - Hierarchical classification, which classifies scheduling methods by a multilevel tree.
  - flat classification based on a single attribute. These include methods based on adaptive versus nonadaptive, load balancing, bidding, or probabilistic and one-time assignment versus dynamic reassignment.

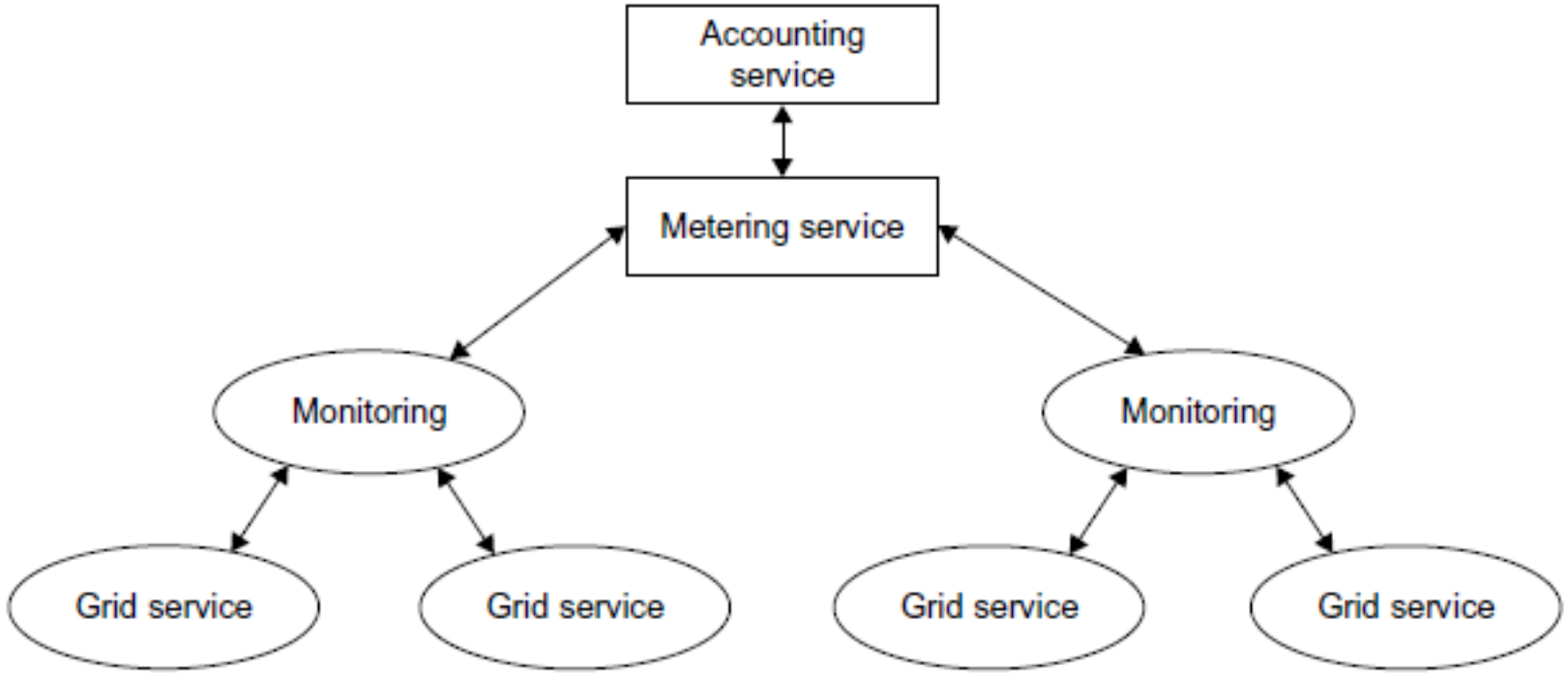


GRID RESOURCE MANAGEMENT AND BROKERING

Service Accounting and Economy Model

Economics-based resource trading is an effective way to allocate grid resources.

- First, resource providers register their shared resources together with resource information in a grid market. The resource information may include hardware capacity, software configuration, and price policy.
- Then resource customers submit their jobs with special requirements to a resource broker. These requirements may include hardware capacity, software configuration, QoS, and budget.
- Next, the broker searches and allocates qualified resources in the grid market for each job. When the jobs are successfully completed, the customers are charged for the resources allocated.



GRID RESOURCE MANAGEMENT AND BROKERING

Monitoring and Metering Services

- This component directly collects raw usage information (e.g., CPU time) from grid services or resources. Generally speaking, each grid service container or computing node should have a monitoring component deployed.
- The collected raw data is forwarded to the upper-level component, the metering component.
- The raw data can be gathered by querying the grid service, analyzing the log data of the application, or obtaining it directly from the underlying system.
- The metering service obtains the raw usage information from the monitoring component, and applies charge policies based on service usage.
- The charge policies are the mappings between raw usage events and monetary charge events. The charge policies specify the formulas used to obtain the cost from the application, user, and result.
- To maintain good grid economy, the accounting system demands a grid economy model.
- The usage data collected by the accounting system will be considered as the foundation of charges to customers.
- If the usage data is faked or modified by malicious services, incorrect actions will be operated on customers' and providers' accounts.
- As a result, the order of the grid economy will be seriously disturbed. Some accounting systems have taken measures to ensure security.

GRID RESOURCE MANAGEMENT AND BROKERING

Monitoring and Metering Services

- GridBank, the accounting system of Gridbus, uses SOAP over Globus Toolkit sockets.
- Access from the remote service is authenticated and authorized with this grid security infrastructure.
- Grids normally consist of multiple real organizations. The accounting has to be performed across multiple domains,so the accounting system must provide the access interface of standard grid services.
- Globus Toolkit is the most popular and de facto standard for grid services.

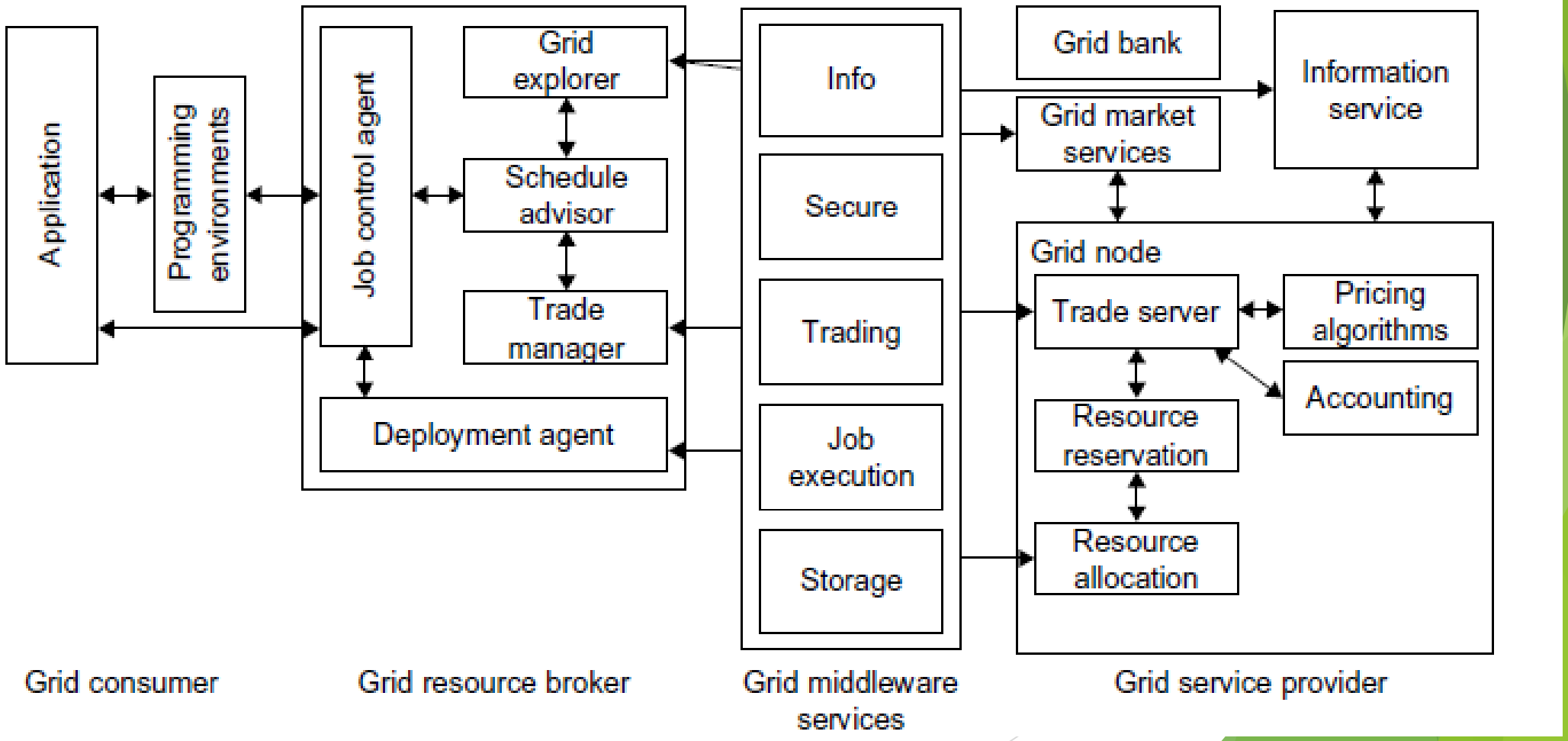
Accounting Component

- The primary function of an accounting module is to maintain the economic relationship between resource customers and providers.
- By receiving and converting charge events from the metering module, the accounting module records billing events.
- An accounting event is the action which will be operated on the account of each grid user.
- The account of the grid user holds the information on the balance amount. Some other features are also demanded for accounting in a grid, such as the security of the accounting system.

GRID RESOURCE MANAGEMENT AND BROKERING

Resource Brokering with Gridbus

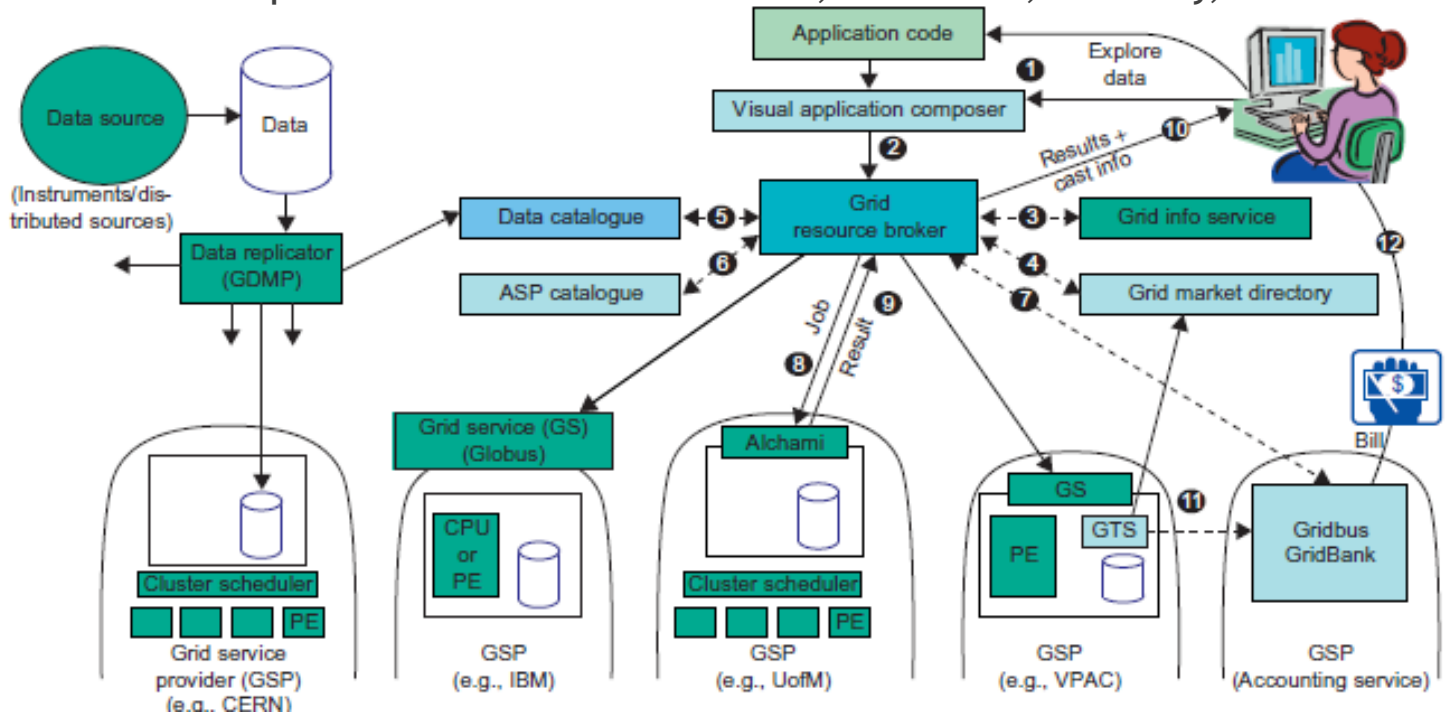
- Grid environment from an operations perspective with the components organized according to their deployment and functionality.



GRID RESOURCE MANAGEMENT AND BROKERING

Resource Brokering with Gridbus

- To make resources constituents of the grid, they need to be accessible from different management domains. This can be achieved by installing core grid middleware such as Globus in UNIX/Linux environments. Multinode clusters need to be presented as a single resource to the grid, and this can be achieved by deploying job management systems such as the Sun Grid Engine on them.
- In a grid environment where data needs to be federated for sharing among various interested parties, data grid technologies such as SRB, Globus RLS, and EU DataGrid need to be deployed. The user-level middleware needs to be deployed on resources responsible for providing resource brokering and application execution management services. Users may even access these services via web portals. Several grid resource brokers have been developed. Some of the more prominent include Nimrod-G, Condor-G, GridWay, and Gridbus Resource Broker.



GRID RESOURCE MANAGEMENT AND BROKERING

Resource Brokering with Gridbus

- The following 11 steps are followed to aggregate the Grid resources:

1. The user composes his application as a distributed application (e.g., parameter sweep) using visual application development tools.
2. The user specifies his analysis and QoS requirements and submits them to the grid resource broker.
3. The grid resource broker performs resource discovery using the grid information service.
4. The broker identifies resource service prices by querying the grid market directory.
5. The broker identifies the list of data sources or replicas and selects the optimal ones.
6. The broker identifies computational resources that provide the required services.
7. The broker ensures that the user has necessary credit or an authorized share to utilize the resources.
8. The broker scheduler analyzes the resources to meet the user's QoS requirements.
9. The broker agent on a resource executes the job and returns the results.
10. The broker collates the results and passes them to the user.
11. The meter charges the user by passing the resource usage information to the accountant.

Grid Application Trends and Security Measures

Grid Workload and Performance Prediction

- Grid performance is directly related to the collective workload to be executed on a large number of processors in participating grid sites.
- Predicting the collective grid workload is a very challenging task, because heterogeneous resources are highly distributed under the control of different organizations.
- The grid workload is represented by a collective load index among all processors. The load index X(t) is the percentage of processors utilized within a unit time interval [t − 1, t]. All discrete time instances t are denoted by non-negative integers.
- For simplicity, assume five minutes per time step. Load index reflects the CPU utilization rate among all processors in a grid. For example, X(t) = 0.45 implies that 45 percent of the processors are busy during the observation period.
- In general, the load index is used to estimate the percentage of peak performance achievable on a given computational grid. The management console of such a grid can monitor CPU utilization.
- Workload managers in large grid infrastructures are notoriously weak in determining correct scheduling scenarios, which affects the application execution time on the grid.

Grid Application Trends and Security Measures

Grid Workload and Performance Prediction

- **Adaptive Workload Prediction -** Time series of events
- Auto-Regression (AR) Method
- H-Model for Workload Prediction
- Adaptive Prediction Scheme (AH-Model)

Grid Application Trends and Security Measures

Trust Models for Grid Security Enforcement

- Many potential security issues may occur in a grid environment if qualified security mechanisms are not in place. These issues include network sniffers, out-of-control access, faulty operation, malicious operation, integration of local security mechanisms, delegation, dynamic resources and services, attack provenance, and so on.
- The first challenge: integration with existing systems and technologies. The resources sites in a grid are usually heterogeneous and autonomous. It is unrealistic to expect that a single type of security can be compatible with and adopted by every hosting environment. At the same time, existing security infrastructure on the sites cannot be replaced overnight. Thus, to be successful, grid security architecture needs to step up to the challenge of integrating with existing security architecture and models across platforms and hosting environments.
- The second challenge: interoperability with different “hosting environments.” Services are often invoked across multiple domains, and need to be able to interact with one another. The interoperation is demanded at the protocol, policy, and identity levels. For all these levels, interoperation must be protected securely. The third challenge is to construct trust relationships among interacting hosting environments. Grid service requests can be handled by combining resources on multiple security domains. Trust relationships are required by these domains during the end-to-end traversals. A service needs to be open to friendly and interested entities so that they can submit requests and access securely.

Grid Application Trends and Security Measures

Trust Models for Grid Security Enforcement

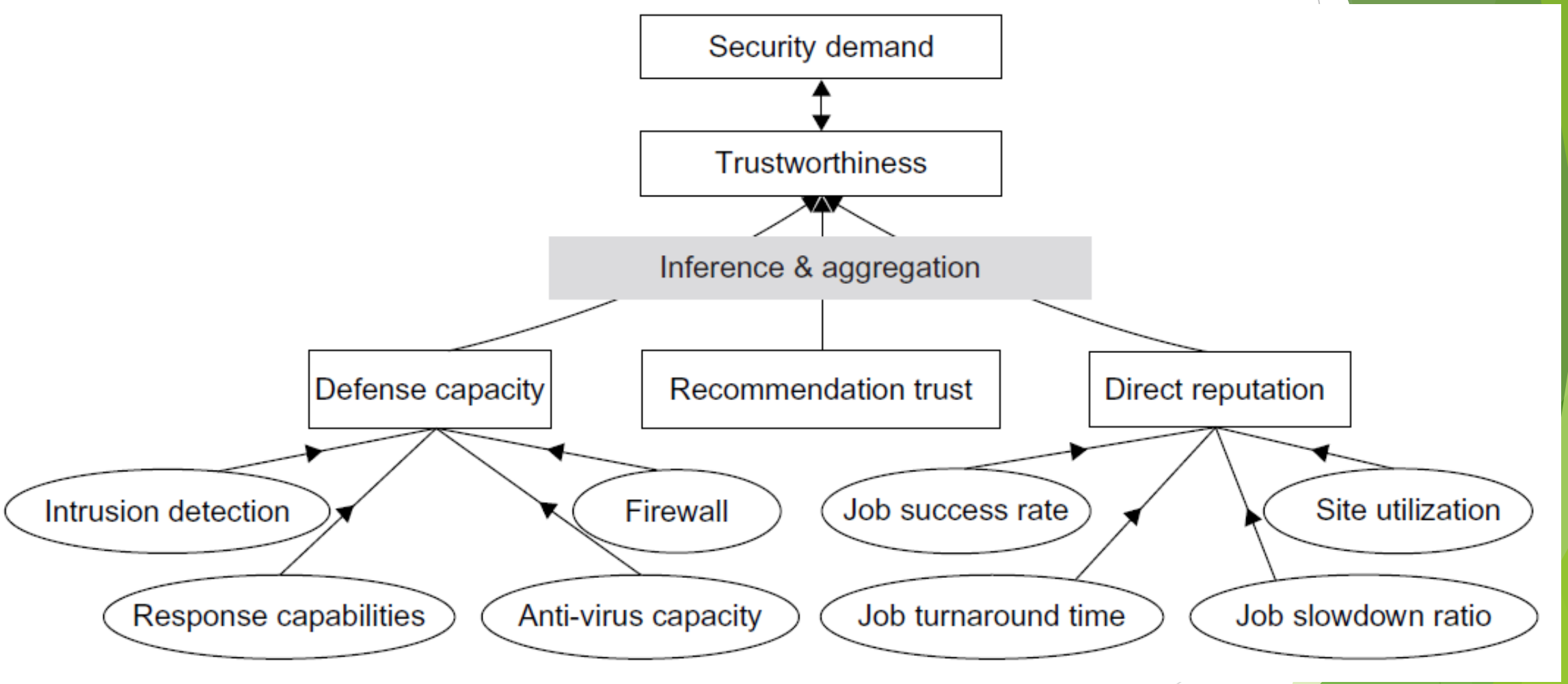
- A trust relationship must be established before the entities in the grid interoperate with one another.
- The entities have to choose other entities that can meet the requirements of trust to coordinate with.
- The entities that submit requests should believe the resource providers will try to process their requests and return the results with a specified QoS.
- To create the proper trust relationship between grid entities, two kinds of trust models are often used.
  - PKI-based model
  - Reputation-based model.

Grid Application Trends and Security Measures

A Generalized Trust Model

- An inference module is required to aggregate these factors.
- Followings are some existing inference or aggregation methods.
- An intra-site fuzzy inference procedure is called to assess defense capability and direct reputation.
- Defense capability is decided by the firewall, intrusion detection system (IDS), intrusion response capability, and anti-virus capacity of the individual resource site.
- Direct reputation is decided based on the job success rate, site utilization, job turnaround time, and job slowdown ratio measured.
- Recommended trust is also known as secondary trust and is obtained indirectly over the grid network.

Grid Application Trends and Security Measures



Grid Application Trends and Security Measures

Reputation-Based Trust Model

- jobs are sent to a resource site only when the site is trustworthy to meet users' demands.
- The site trustworthiness is usually calculated from the following information:
- the defense capability, direct reputation, and recommendation trust.
- The defense capability refers to the site's ability to protect itself from danger. It is assessed according to such factors as intrusion detection, firewall, response capabilities, anti-virus capacity, and so on.
- Direct reputation is based on experiences of prior jobs previously submitted to the site.
- The reputation is measured by many factors such as prior job execution success rate, cumulative site utilization, job turnaround time, job slowdown ratio, and so on.
- A positive experience associated with a site will improve its reputation.
- On the contrary, a negative experience with a site will decrease its reputation.

Grid Application Trends and Security Measures

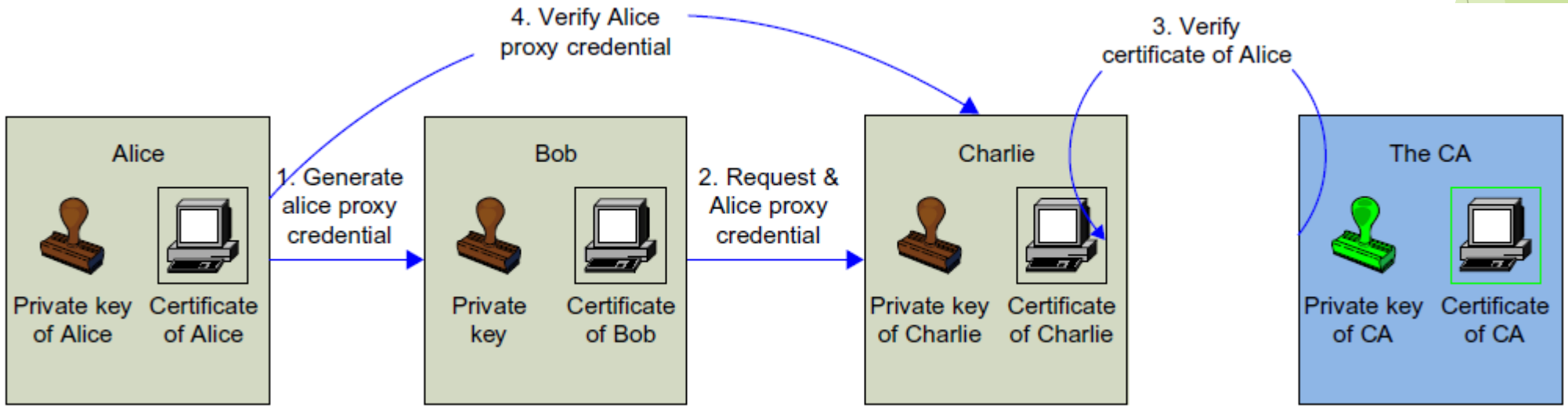
A Fuzzy-Trust Model

- The job security demand (SD) is supplied by the user programs. The trust index (TI) of a resource site is aggregated through the fuzzy-logic inference process over all related parameters. Specifically, one can use a two-level fuzzy logic to estimate the aggregation of numerous trust parameters and security attributes into scalar quantities that are easy to use in the job scheduling and resource mapping process.
- The TI is normalized as a single real number with 0 representing the condition with the highest risk at a site and 1 representing the condition which is totally risk-free or fully trusted. The fuzzy inference is accomplished through four steps: fuzzification, inference, aggregation, and defuzzification. The second salient feature of the trust model is that if a site's trust index cannot match the job security demand (i.e., SD > TI), the trust model could deduce detailed security features to guide the site security upgrade as a result of tuning the fuzzy system.

Grid Application Trends and Security Measures

Authentication and Authorization Methods

- The major authentication methods in the grid include passwords, PKI, and Kerberos. The password is the simplest method to identify users, but the most vulnerable one to use. The PKI is the most popular method supported by GSI. To implement PKI, we use a trusted third party, called the certificate authority (CA). Each user applies a unique pair of public and private keys. The public keys are issued by the CA by issuing a certificate, after recognizing a legitimate user. The private key is exclusive for each user to use, and is unknown to any other users. A digital certificate in IEEE X.509 format consists of the user name, user public key, CA name, and a secrete signature of the user. The following example illustrates the use of a PKI service in a grid environment.



Grid Application Trends and Security Measures

Authentication and Authorization Methods

- Authorization for Access Control
- Three Authorization Models

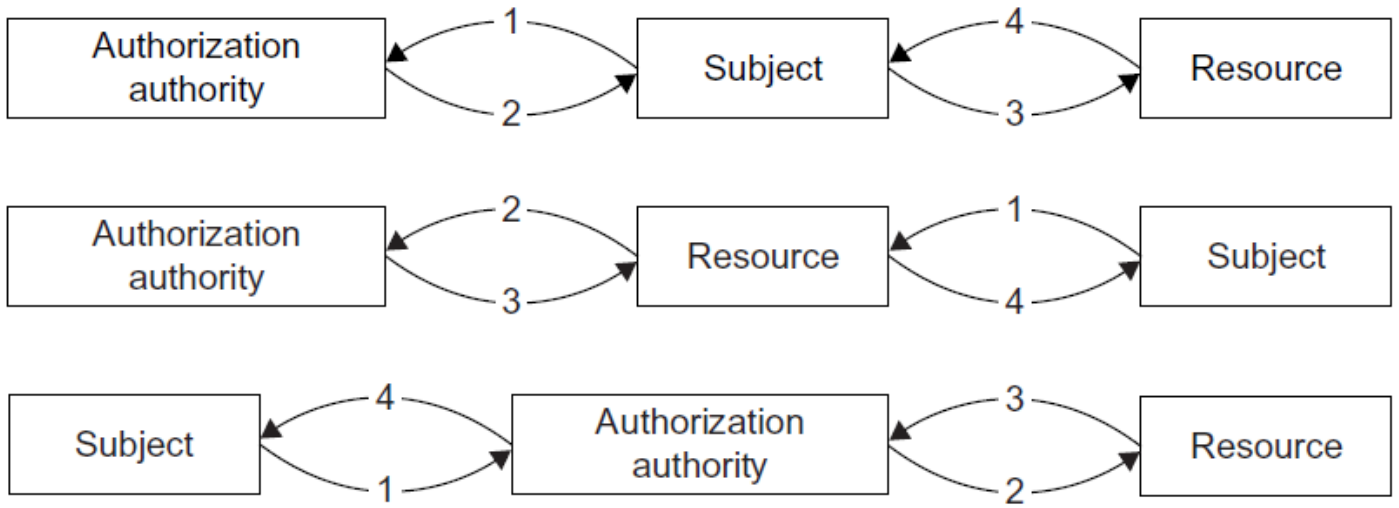


FIGURE 7.31

Three authorization models: the subject-push model, resource-pulling model, and the authorization agent model.

Grid Application Trends and Security Measures

Grid Security Infrastructure (GSI)

The grid requires a security infrastructure with the following properties:

- Easy to use.
- Conforms with the VO's security needs while working well with site policies of each resource provider site.
- Provides appropriate authentication and encryption of all interactions.
- The GSI is an important step toward satisfying these requirements. As a well-known security solution in the grid environment, GSI is a portion of the Globus Toolkit and provides fundamental security services needed to support grids, including supporting for message protection, authentication and delegation, and authorization. GSI enables secure authentication and communication over an open network, and permits mutual authentication across and among distributed sites with single sign-on capability. No centrally managed security system is required, and the grid maintains the integrity of its members' local policies. GSI supports both message-level security, which supports the WS-Security standard and the WS-SecureConversation specification to provide message protection for SOAP messages, and transport-level security, which means authentication via TLS with support for X.509 proxy certificates.

Grid Application Trends and Security Measures

GSI Functional Layers

- GT4 provides distinct WS and pre-WS authentication and authorization capabilities. Both build on the same base, namely the X.509 standard and entity certificates and proxy certificates, which are used to identify persistent entities such as users and servers and to support the temporary delegation of privileges to other entities, respectively. As shown in Figure 7.32, GSI may be thought of as being composed of four distinct functions: message protection, authentication, delegation, and authorization.
- TLS (transport-level security) or WS-Security and WS-Secure Conversation (message-level) are used as message protection mechanisms in combination with SOAP. X.509 End Entity Certificates or Username and Password are used as authentication credentials. X.509 Proxy Certificates and WS-Trust are used for delegation. An Authorization Framework allows for a variety of authorization schemes, including a “grid-mapfile” ACL, an ACL defined by a service, a custom authorization handler, and access to an authorization service via the SAML protocol. In addition, associated security tools provide for the storage of X.509 credentials (MyProxy and Delegation services), the mapping between GSI and other authentication mechanisms (e.g., KX509 and PKINIT for Kerberos, MyProxy for one-time passwords), and maintenance of information used for authorization (VOMS, GUMS, PERMIS).

Grid Application Trends and Security Measures