Data modelling in Grid Information Services

Grid Information Services Workshop
Edinburgh, 24th April 2003

Peter Gietz, DAASI International GmbH
Peter.gietz@daasi.de

DAASI International
Directory Applications
for Advanced Security
and Information Management

AGENDA

- The Problem
  - Definitions
  - tasks
- The solutions
  - Technologies and their information model
    - LDAP
    - RDBMS
    - XML/OGSA
  - Data modelling
- Where to go from here

What is a Grid Information Service?

- An Information service that enables Grid computing
- Old Definitions for Grid Computing:

„*The Grid is a consistent and standardized environment for collaborative, distributed problem solving that requires high performance computing on massive amounts of data that are stored, and/or generated at high data rates using widely distributed, heterogeneous resources* „

„*The Grid is an inherently layered architecture that provides for common services and a diversity of middleware that supports building distributed, large-scale, and high performance applications and problem solving systems.* „

(W.E. Johnston as quoted by Ian Foster)

- A Grid Information Service is a large-scale distributed middleware that provides information about resources in the net for virtual organisations (a set of individuals and/or institutions defined by a set of sharing rules)

What kind of Grids?

- Computational Grid
  - Middleware that enables software applications to integrate instruments, displays, computational resources (CPUs, harddisks, etc.)
- Information Grid
  - Middleware that enables software applications to get information about resources, data and the rest
- Knowledge Grid
  - Middleware that enables software applications to integrate knowledge (=relations between concepts and information)

The Task

- Distribution of data and computing ressources in broadbandwith networks to be able to provide petabyte storage and petaflops computing power
- Promotion of international collaboration
- Optimal utilization of resources (storage, CPUs, measuring devices, experimental devices
- All this was done before and was called metacomputing
- New in „Grids“ is the concept of standardised interface to meta computing, the so called Middleware.
- The Global Grid Forum (GGF) took up the task to create such standards in an IETFish way

What exactly do we want to do?

- Find a number of CPUs to perform a batchjob
- Find harddisk storage to store many and big files
- Have these files replicated in the net and find a copy of a file
- Find network bandwidth for real time visualisation of huge data masses
- Monitor the network
- Allocate ressources in advance

The Tools

- Globus Toolkit 2 has:
  - Globus Resource Allocation Manager (GRAM)
  - Grid Security Infrastructure (GSI)
  - Grid Resource Information Protocol (GRIP)
  - Grid File Transfer Protocol (GridFTP)
  - Metacomputing Directory Service (MDS)
  - Global Access to Secondary Storage (GASS)
  - ...
- Based on standards LDAP, GSS-API, X.509

LDAP was the first approach

- Lightweight Directory Access Protocol
- Globus Metacomputing Directory Service (MDS)
  - Based on LDAP (OpenLDAP 2.x.x)
  - Directory distributed on a collection of LDAP servers
  - Can be updated by
    - Information providers and tools
    - Applications and usersBackend tools
  - The information is dynamically available
- Two classes of MDS servers:
  - Grid Resource Information Service (GRIS)
  - Grid Index Information Service (GIIS)

Grid Resource Information Service

- (Dynamic) Information about specific resources:
  - Load, process information, storage information, etc.
- Supports multiple information providers
- Well known port 2135
- Answers questions like:
  - How much memory does machine have?
  - Which queues on machine allows large jobs?

Grid Index Information Service (GIIS)

- Describes a class of servers
  - Supplies collection of information gathered from multiple GRIS servers
  - Each GIIS is optimized for particular queries, e.g.:
    - Which Solaris machines have >16 CPUs?
    - Which storage servers have >100Mbps bandwidth to host X?
- More efficient queries than talking to each single GRIS server

Replica management

- Maintain a mapping between logical names for files and collections and one or more physical locations
- replica cataloging and reliable replication as two fundamental services
  - LDAP is used as catalog format and protocol, for consistency
  - LDAP object classes for representing logical-to-physical mappings in an LDAP catalog

Why LDAP?

- It is a database or information model (X.500)
  - Hierarchical structure
  - Object oriented
  - Extensible for any kind of data
- It is a Client/server network protocol
  - Internet standard
  - Inbuilt security (MD5 challenge response, TLS)
  - Allows for distribution of data in the net (just like WWW!)
  - Allows for replication of the data in the net
  - Flexibly extensible

LDAP Information Model 1

- Data are stored in entries
- Entries are ordered as tree nodes
- In the Directory Information Tree (DIT)
  - Every node has 0 to n children nodes
  - Every node except root has 1 parent node
- An entry has a distinguished name
  - in its hierarchy level: Relative Distinguished Name (RDN)
  - all RDNs on the path from root form the Distinguished Name (DN)
- No two siblings, i.e. entries with a common parent can have the same RDN
- Thus no two entries in the whole Directory can have the same DN

LDAP Information Model 2

- An Entry contains a number of Attributes
- An Attribute consists of:
  - Attribute Type
  - Attribute Value(s)
- An Attribute Type has an associated Attribute Syntax
- The Attribute Value has to conform to that syntax
- Matching Rules to compare Attribute values for
  - equality
  - substring
  - ordering
  - extensible (selfdefined) matching

LDAP Information Model 3

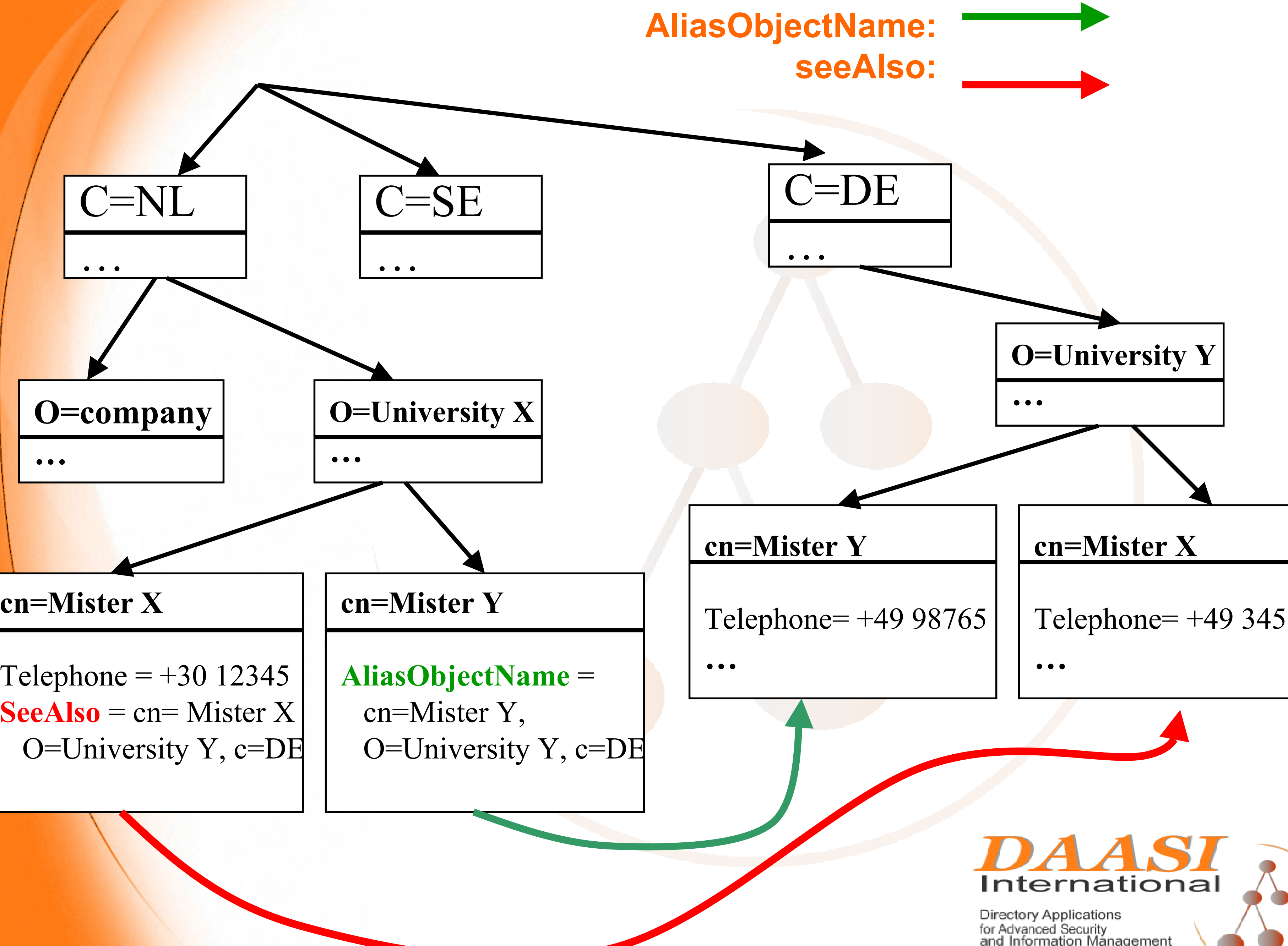
- One or more Attribute type/value pairs form the RDN
  - The Naming Attributes or
  - The Distinguished Attributes
- An Entry must have one or more Objectclass Attributes which:
  - Characterizes the Entry, e.g. Person
  - Defines a set of usable Attributes the entry may contain and must contain
- Objectclasses can inherit Attributes from other Objectclasses
- A set of Objectclasses, Attributes and Syntaxes for a special purpose is called schema

LDAP Information Model 3

- You can define your own:
  - Object Classes
  - Attribute Types
  - Attribute Syntaxes
  - Matching Rules
- You can locally use self defined schemas
- If you want them to be used globally you have to
  - standardize them (IETF)
  - or at least register them

LDAP functional model

- Authentication and control operations:
  - bind
  - unbind
  - abandon
- Interrogation operations:
  - search
  - compare
- Update operations:
  - add
  - delete
  - modify
  - modifyDN


AliasObjectName:
seeAlso:
C=NL
...
C=SE
...
C=DE
...
O=company
...
O=University X
...
O=University Y
...
cn=Mister X
Telephone = +30 12345
SeeAlso = cn= Mister X
O=University Y, c=DE
cn=Mister Y
AliasObjectName =
cn=Mister Y,
O=University Y, c=DE
cn=Mister Y
Telephone= +49 98765
...
cn=Mister X
Telephone= +49 345
...

LDAP Search Parameters

1. base object or base DN
   - where in the DIT the search starts
2. scope
   - base (read the entry specified by the base dn)
   - onelevel (search only in the hierarchical level of the basedn)
   - subtree (search in level of base DN and below)
3. derefAliases
   - neverDerefAlias (do not dereference aliases in searching or in locating base object)
   - derefInSearching (dereference only in subordinates of base object)
   - derefFindingBaseObject (dereference only in locating the base object)
   - derefAlways (dereference aliases in searching subordinates and in locationg base object)

LDAP Search Parameters

4. size limit
   - limit the number of entries to get back
5. time limit
   - limit the time the server should spend to fulfil the request
6. attrsOnly
   - Boolean. If set to true only the attributenames will be sent back, not the values
7. Filter
   - expression that describes the entries to be returned
8. attributes
   - a list of comma separated attributes Types to be returned
   - e.g.: cn, telephonenumber
   - can be specified by OID as well, e.g. 2.5.4.3, 2.5.4.20
   - * means all user attributes
   - 1.1 (there is no such attribute OID) for no attributes

LDAP Search Filter Operators

- Equality
  - Only for attributes with equality matching rule
  - e.g.: (cn=Mister X) only entries with common name equals “Mister X”
- Negation operator
  - e.g. (!(cn=Mister X)) all entries but the one with cn equals “Mister X”
- Substring
  - Only for attributes with substring matching rule
  - e.g. (cn=Mister*) all entries with cn beginning with “Mister”
- Approximate
  - Implementation dependent
  - e.g.: (cn~=Mister) all entries with cn sounding similiar to “Mister”

LDAP Search Filter Operators

- Greater than or equal to and less than or equal to
    - Only for attributes with ordering matching rule
    - e.g. (sn<=Smith) all entries where sn equals “Smith” or is lexicographically above “Smith” (from sn=Adam to sn=smirnow)
    - (age>21) is not possible, use (!(age<=21)) instead
- Presence
    - e.g. (telephoneNumber=*) all entries that contain a telephone number
    - e.g. (objectclass=*) all entries, since every entry contains at least one objectclass

Search filter combinations

- Filters can be combined
  - AND operator: &
    - e.g. (& (cn=Mister X) (mail=*dot.com)) only entries that have both cn=Mister X and a mail address ending with dot.com
  - OR operator: |
    - e.g.: (| (cn=Mister X) (sn=Xerxes)) all entries that have cn=Mister X or sn=Xerxes

LDAP search filter extensions

- LDAPv3 defines an extensible matching filter
  - syntax: attr [“:dn”] [“:” matchingrule] “:=“ value
    - attr is an attribute name
    - “:dn” says that also the attribute in the dn should be searched as well
    - matching rule given by an OID or associated descriptive name
  - examples:
    - (cn:1.2.3.4.5.6:=Mister X) use matching rule 1.2.3.4.5.6 for comparision
    - (o:dn:=company) search for o=company in attributes and also in DN

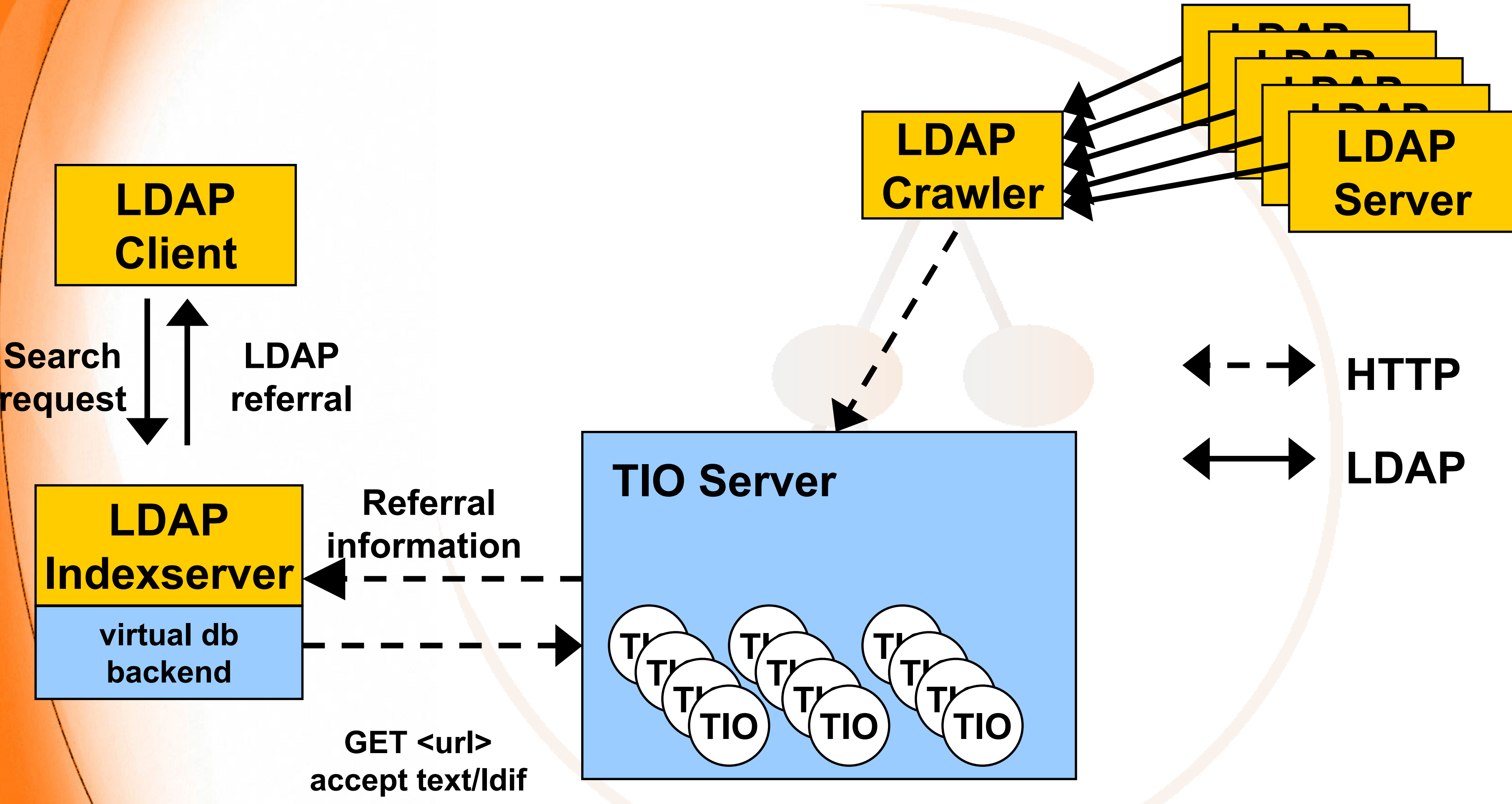
LDAP URL (RFC 2255)

- Format:
  - ldap://<host>:<portnumber>/<basedn>?<attrlist>?<scope>?<filter>?<extensions>
- Example:
  - ldap://myhost.org:9999/c=NL,o=University?cn,telephonenumber?subtree?(cn=Mister X)

Advantages of LDAP

- Objectoriented Datamodel
- Open Standard thus vendor independentness
- Distribution enables high scalability
- Replication enables high reliability
- High security through granular access control and secure authentication mechaqnisms
- Data access via standardised TCP/IP based network protocol
- The same data can be used by different applications
- There is a stable open source implementation (OpenLDAP)

LDAP Index system



Why not LDAP?

- „inflexible data model“
  - An entry is an entry is an entry
  - The information model intends to map reality
  - Thus you have an entry for a person and a different entry for a computer
  - Relations are a bit more complicated, but doable (see below)
- „poor query language“
  - Cannot give results based on computation on two different objects in the struture
  - No Join operation
- „You have to anticipate questions“

XML the new direction in GGF

- OGSA/OGSI: Open Grid Services Architecture/Infrastructure
  - Common framework for Grid Services
  - Every OGSI-compliant service can be used and managed via common interfaces
  - Modeled after the new paradigm: „Web Services“ which is the new industry standard
  - „everything is a service“

Web Services

- Service description via WSDL (Web Service Description Language)
- Service Registration via UDDI (Universal Description Discovery & Integration)
- Communication via SOAP (Simple Object Access Protocol)

Grid Service

- A Grid Service provides a set of well-defined interfaces that follow specific conventions
  - Discovery
  - Dynamic service creation
  - Lifetime management
  - Notification
  - Manageability
- Can also be transient
  - A query against a data base
  - A running data transfer
  - Advance reservation

Information Service in Globus Toolkit 3

- New Index Service
  - Replacement for MDS
  - Interface for connecting external information provider
  - Generic framework for aggregation of service data
  - A registry of Grid Services
  - Queries either simple „by name“ or with more complex languages like XPath

Problems with XML?

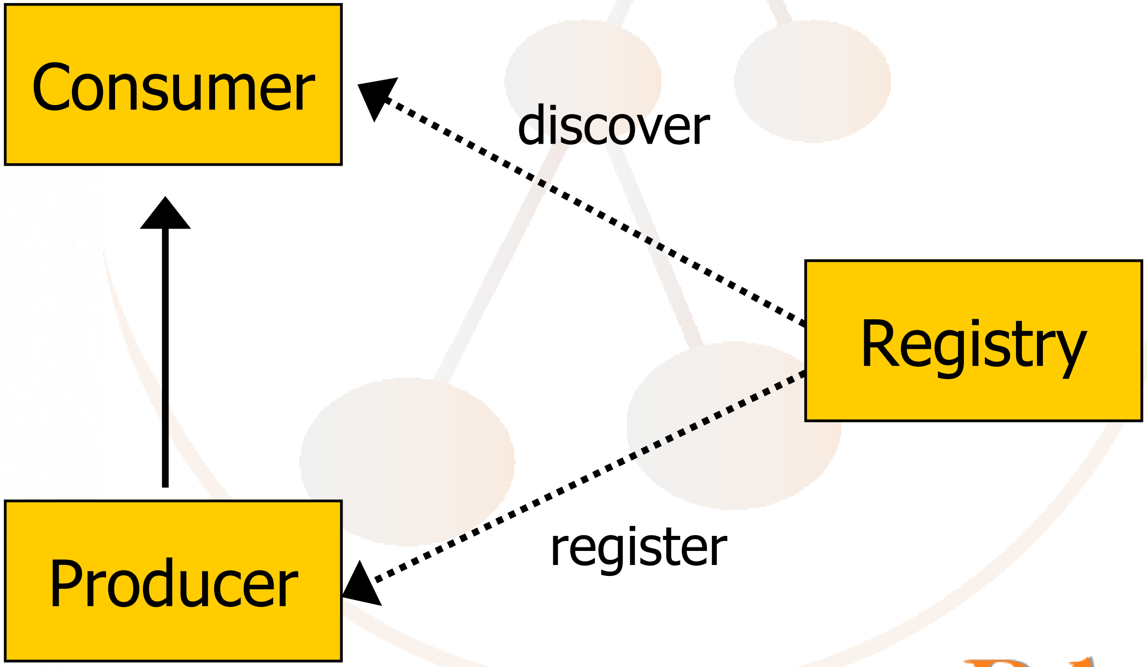
- Do we have a highly scalable XML-Database?
- Do XML-Databases provide granular access control?
- What about web services security?
- Do we have a standardised XML query language?
  - Xpath, Xquery, XQL, ...

RDBMS as alternative?

- Relational model
- Data organised in tables
- A table is a set of objects of the same type
- A single query can combine information scattered over many different tables
- Transactions support a „undo“
- But RDBMS are not suited to be distributed in the net (not true anymore)

R-GMA of the DataGrid Project

- Relational Grid Monitoring Architecture
- Implementation of the GGF GMA with RDBMS technology



R-GMA

- Separates description of a table (the schema) from list of providers of that table (the registry)
- Registry holds a table identifier for each logical table
- Defines an XML based protocol over HTTPs containing SQL statements
- HTTP Servlet technology
- Tables can be distributed in the net

Data modelling

- We have a variety of technologies to use (XML DB, RDBMS, LDAP)
- We want to store the same data (computing elements, storage elements, network elements)
- We should be able to use the same schema
- We need an independent data modelling technology

GGF Information Services Area Group Charter:

- This area group tries to identify requirements for and facilitate the development of interoperable models and mechanisms for the information services necessary for doing grid-based computing. This includes
  1. The definition of meta-models to allow the creation of interchangeable schemas.
  2. The definition of formalisms to describe such schemas uniformly by the various working groups.
  3. The definition of a mechanism to access information that is stored in the schema.
  4. The educational outreach in order to provide the working groups with the necessary knowledge to use the models, the formalisms, and the mechanisms.
  5. The verification of the feasibility of the models, formalism, and mechanisms …

Grid Object Specification language

- First attempt for technology independent schema specification
- Enabled mappings to XML and LDAP
- Was it ever really used by other Grid WGs?
- Several versions (GOSv3 never made it)

GLUE schema

- Grid Laboratory Uniform Environment (GLUE)
- Aim to provide interoperability between EU and US HEP Grid projects
- Common schemas to describe Grid resources
  - Compute Elements
  - Storage Elements
  - Network Elements
- Specification in UML
- Mappings to LDAP (MDS), XML, and SQL (R-GMA)

DMTF CIM

- Common Information Model of the Distributed Management Task Force
- Industry approach for enabling management of real world managed objects
- UML is used to define the structure of the meta schema
- Four parts:
  - Modeling language and syntax
  - Management schema (core, common, extension)
  - Protocol to encapsulate syntax and schema (XML/HTTP)
  - Compliance document

Three Layers of CIM

- Core Model
  - Defines schema applicable to all areas of management
  - E.g.

    ManagedElement
    LogicalDevice
    Service
    ManagedSystemElement
    PhysicalElement
    ServiceAccessPoint
    System
    Collection
    Associations (relationships between schema classes)
- Common Model
  - Defines schemas to particular areas of management
  - Independent of technology or implementation
  - Following models have been defined

    Physical
    Network
    Events
    System
    Application
    Policy
    Devices
    Network
    User & Security
- Extension Schema are technology- or vendor-specific extension to the common model
  - Only Core + Common model belong to the CIM standard schema

CIM Meta schema

- The lego bricks:
  - Classes
  - Properties
  - Methods
  - Associations
  - Triggers
  - Indications
  - Qualifiers

Classes

- Describing objects, a type of thing
  - Instances of classes: describing a special object, a thing
- Classinheritance: A subclass inherits qualities of the superclass
- A Class can have:
  - Properties: Attributes of the Class
  - Methods: functions connected with a Class

Properties

- value used to denote a characteristic of a class
- Sort of a pair of functions:
  - one to set the property value
  - one to return the property value.

Methods

- An operation describing the behavior of a class
- A method can have an override relationship with another method from a different class
  - The domain of the overridden method must be a superclass of the domain of the overriding method

Special Class Association

- Associations: defining relationships between Objects
  - Properties of Associations are references pointing to the related classes
  - Special Association is Aggregation having aggregation relationship as property

Special Class: Indication

- An object created as a result of the occurrence of an event
- Event is a status change noticed by a Trigger
- Indication is related with zero or more *triggers* via an association
- Triggers can create instances of the Indication

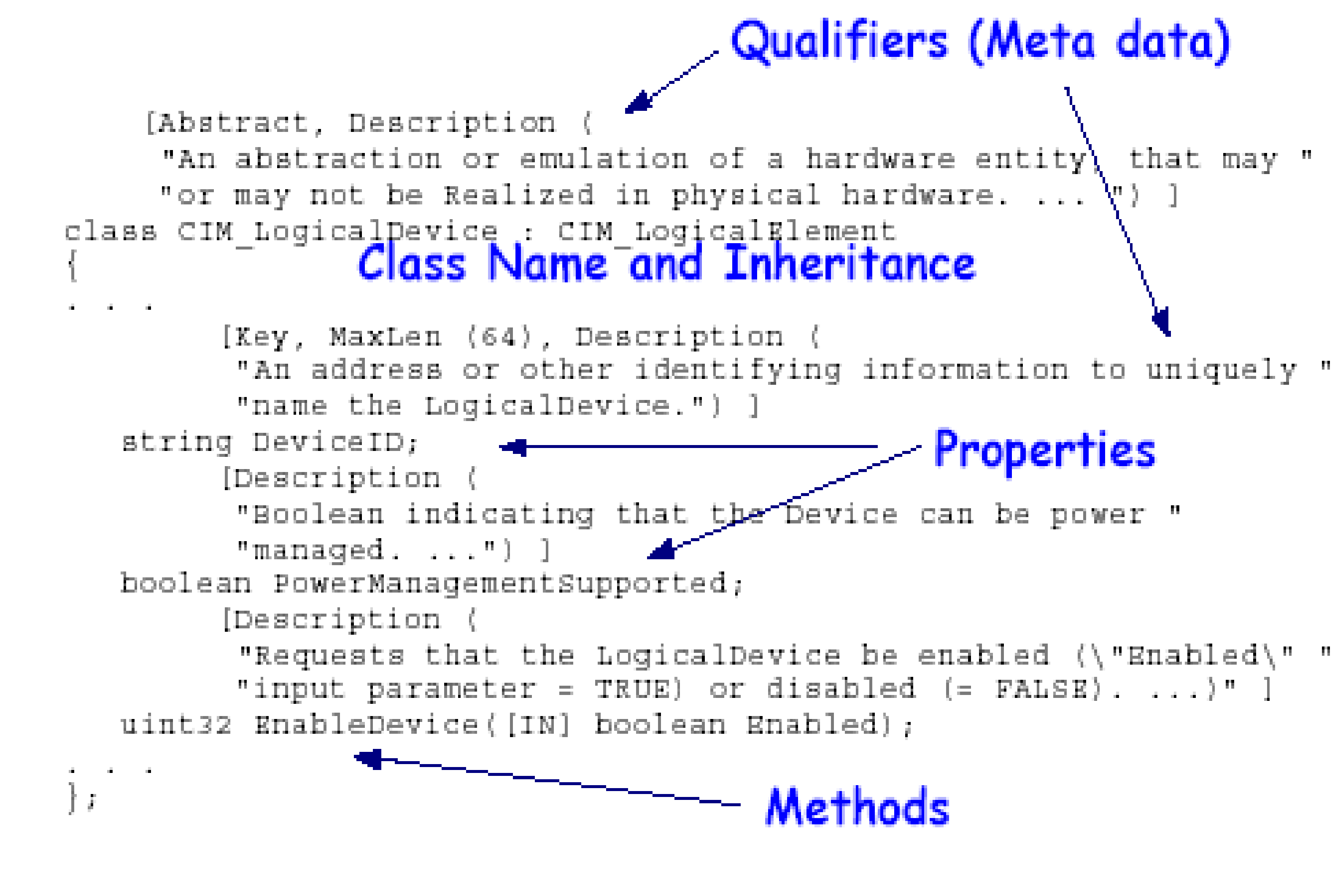
Qualifier

- Metadata to characterize classes, properties, methods and other elements of the schema
- qualifier has:
  - a name
  - a type
  - a value of this type
  - a scope
  - a flavor
    - **EnableOverride, DisableOverride,**
    - **ToSubclass and Restricted,**
  - a default value

MOF

- The primary way of specifying schema in CIM is through the Managed Object Format (MOF)
- A compiled language created by the DMTF and based on the Interface Definition Language (IDL)
- Used to define static or dynamic classes and instances
- C like syntax

Example



```
    [Abstract, Description (
     "An abstraction or emulation of a hardware entity, that may "
     "or may not be Realized in physical hardware. ... ") ]
class CIM_LogicalDevice : CIM_LogicalElement
{
. . .
          [Key, MaxLen (64), Description (
           "An address or other identifying information to uniquely "
           "name the LogicalDevice.") ]
   string DeviceID;
          [Description (
           "Boolean indicating that the Device can be power "
           "managed. ...") ]
   boolean PowerManagementSupported;
          [Description (
           "Requests that the LogicalDevice be enabled (\"Enabled\" "
           "input parameter = TRUE) or disabled (= FALSE). ...)" ]
   uint32 EnableDevice([IN] boolean Enabled);
. . .
};
```

Source: White Paper: Common Information Model (CIM) Core Model, Version 2.4, August 30, 2000

Why CIM?

- A lot of work has been put into the definition of computing elements by the industry (DMTF)
  - Agreement between different vendors
  - A lot of ready to use classes
  - Based on use cases
  - CIM continuously is being expanded
- Don‘t reinvent the wheel
- Technology independant
- Open Source available

Why CIM?

- Designed to provide a common mechanism to describe data about entities for managing computers (and networks)
- Follows the object oriented paradigm
- Capable of defining anything
- Mappings to LDAP, SQL, and XML possible

Cim-based Grid Schema WG

- The idea behind CGS WG was to start a new approach for common schema definition in GGF
- Follow up activity to GOS
- But no new technology, but using an already existing and well established one
- Provide input to the OGSA WGs
- Initially had a broader scope:

CGS WG charter

- The general goal of the CGS-WG is to produce CIM specifications and schemas of classes not yet standardized by the DMTF but urgently needed in the GGF in the frame of OGSA, and to promote these specifications and schemas within the DMTF for their adoption as CIM standards.
- The primary goal of the CGS-WG is to produce a CIM-based "Job Submission Information Model" (JSIM) based on OGSA requirements, and to promote this model within the DMTF for their adoption as CIM standards.

CGS WG charter 2

- Along with this goal the CGS-WG will specify if and how the new CIM classes for representing JSIM should be mapped into XML and LDAP. Future GGF work on CIM will show if these mappings can be reused for other CIM classes. The inherent generalization capabilities will be taken into account.
- The secondary goal of the CGS-WG is to determine which other grid-computing resources are required by the OGSA but still lack a CIM specification and schema -- for development in either this or perhaps a future working group.
- The CGS-WG will liaise with other OGSA-based working groups for their input.

CGS initial charter 3

- In addition, the group may pursue the following goals:
- Exploring the possible use of evolving CIM-based Open Source software in Grid computing.
- Presenting the results of the WG to the other Grid Working Groups.

Current charter aims

- We reduced our scope to get things done quickly wrap up and set up a new WG
- Only aim now is to specify a CIM Job Submission Interface Model
- This is now almost done (the last issues will be solved at GGF 8)
- A second document will provide an XML mapping
- Work on LDAP mapping will also be done

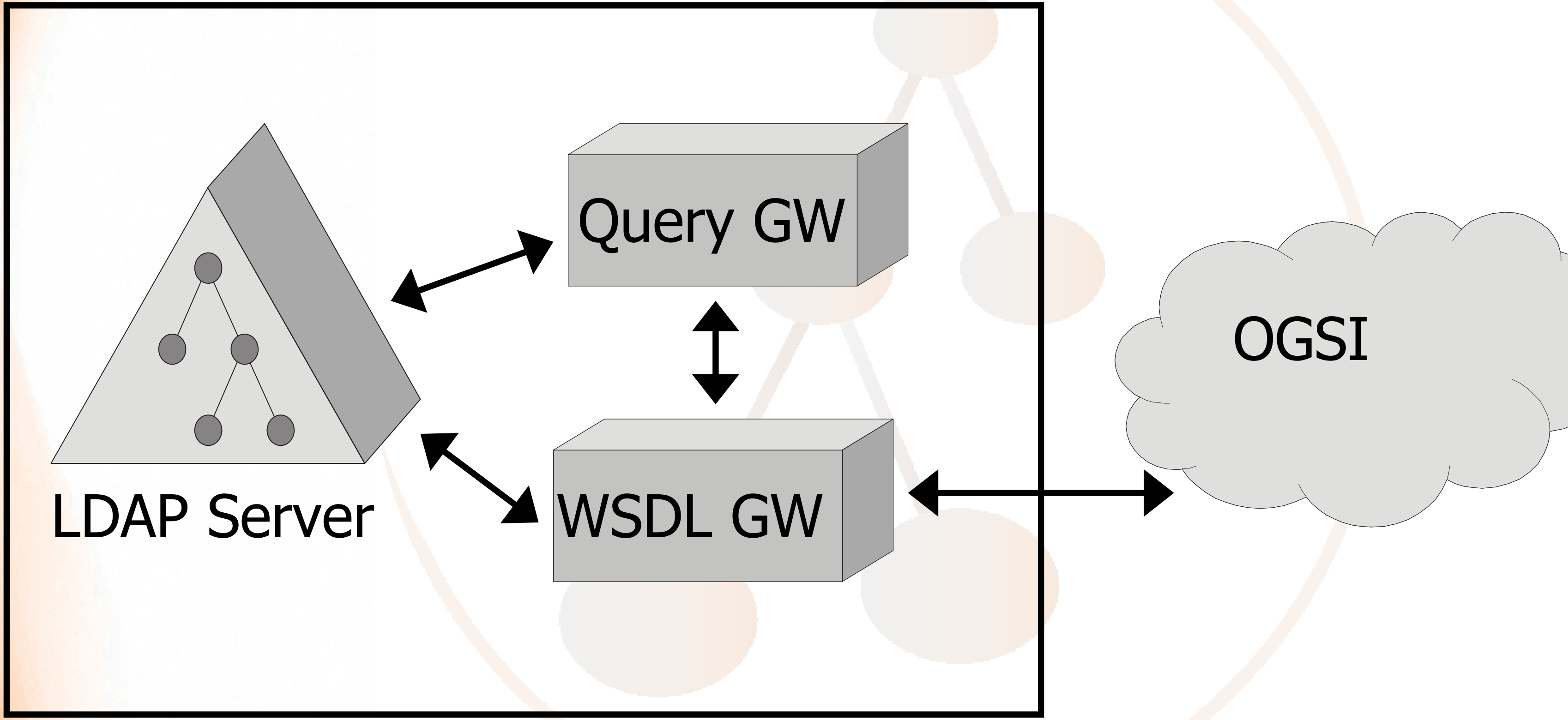
Where to go from here ?

- OGSA based Globus Toolkit 3.0. Will be out this summer
- LDAP based Globus Toolkit 2.2 (released in October 2002) will be supported „at leats through the end of 2003, subject as always to available resources“
- GT3 will be backwards compatible as to the client APIs (GRAM and GridFTP)
- GT2 and GT3 can be run simultaneously on the same resource „with limited interoperability“
- Globus welcomes „suggestions for other ways of assisting migration“

What could we do?

- Is it possible to put an WSDL/OGSI-Gateway in front of an LDAP server, or a whole LDAP indexing system?
- Thus you could integrate your existing infrastructure into the new architecture.
- I would very much like to find out within a research kolaboration
- What about developping an LDAP Gateway for more complex queries (SQL Join)
- But nevertheless: if we model the data technology independent, migration from one technologfy to the other will be easier anyway

Enhanced LDAP



LDAP and UDDI

- Novell has implemented it‘s draft:
  B. Bergeson, K. Boogert: LDAP Schema for UDDI, draft-bergeson-uddi-ldap-schema-01.txt, May, 2002
- eDirectory-based UDDI registry

What about knowledge?

- Metadata
  - Data about information
- Ontologies
  - Concepts and relations between them
  - Computer knows more than inputed:

```
Input: Parents have children
Input: Mother = female parent
Output: Mothers have children
```

Ontologie Storage Proposal

- Combined repository for metadata and ontologies based on LDAP technology and thus accessible with the same protocol
- Large scalability by setting up an Indexing system based on Common Indexing Protocol (CIP)
- Ontologie data model based on CIM which provides a model for associations that can be used for mapping the relations between objects

What could you store?

- Multiple ontologies with links between different ontologies
- General ontologies (e.g. WordNet)
- Special ontologies (e.g. on special subjects)
- Any kind of relations can be defined with CIM and mapped to LDAP
- LDAP provides:
  - Object Class inheritance
  - Attribute inheritance
- Associations and aggregations can be mapped by object classes

Thank you for your attention

- DAASI International
  - http://www.daasi.de
  - Peter.gietz@daasi.de