Naming

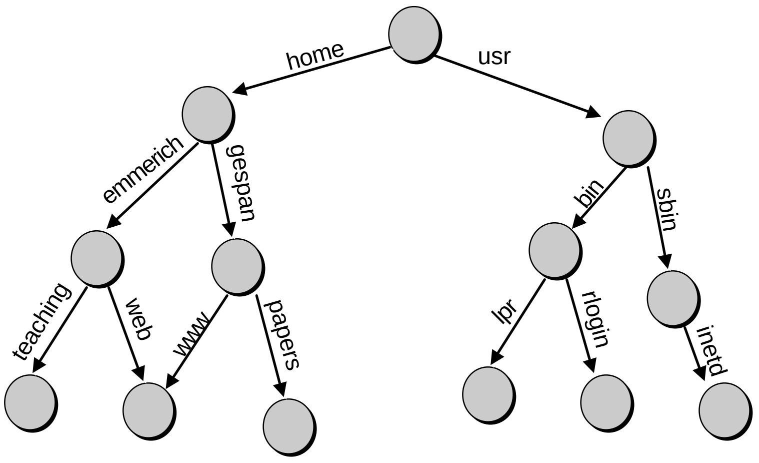
Location Transparency

- ***Avoid using physical locations for locating components!***
- ***Naming:***
  - ***Locating components by external names***
  - ***Similar to white pages***
- ***Trading:***
  - ***Locating components by service characteristics***
  - ***Similar to yellow pages***

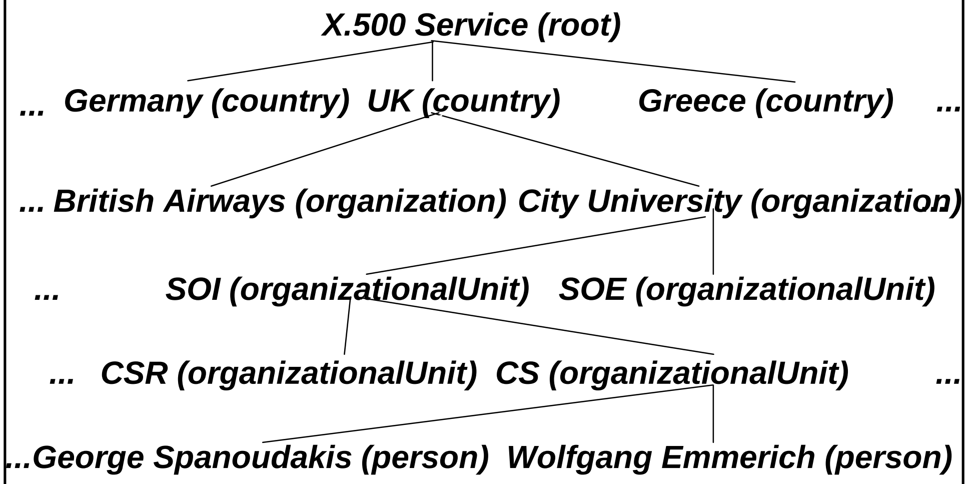
Name Server Examples

- Network File Systems
- X.500 Directory Service
- Internet Domain Name Service
- CORBA Naming Service
- Java Registry

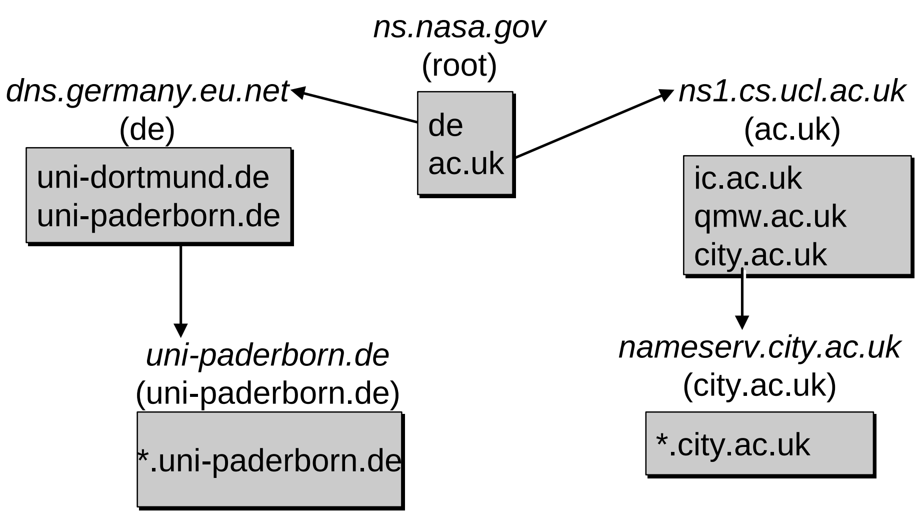
2.1 NFS Directories



X.500 Directory Service


X.500 Service (root)
... Germany (country) UK (country) Greece (country) ...
... British Airways (organization) City University (organization) ...
... SOI (organizationalUnit) SOE (organizationalUnit)
... CSR (organizationalUnit) CS (organizationalUnit) ...
...George Spanoudakis (person) Wolfgang Emmerich (person)

Internet Domain Name Service



Common Characteristics

Concerns for a naming service:

- ***Names.***
- ***Namespaces.***
- ***Naming service provides operations for***
  - ***defining names of components (bind).***
  - ***lookup components by name (resolve).***
- ***Persistence of bindings.***

Common Characteristics

Qualities of service:

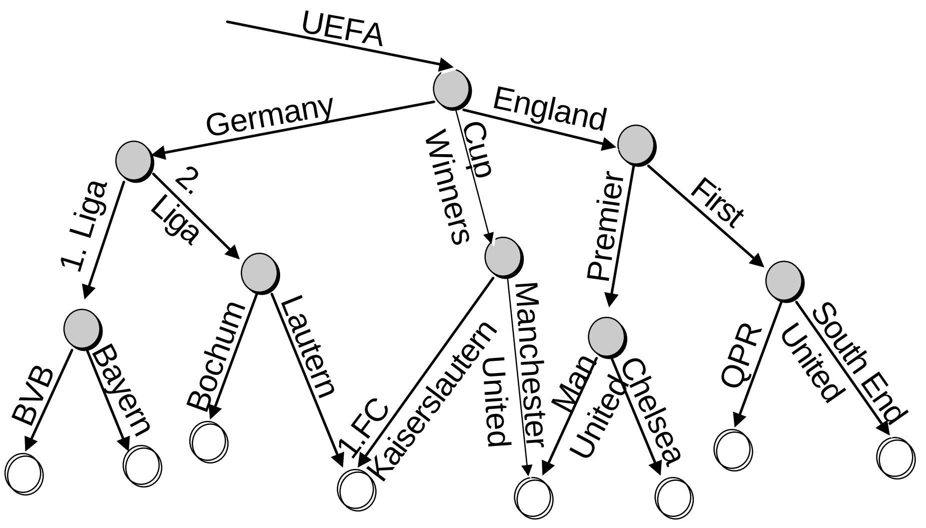
- ***Distribution of name spaces***
- ***Performance profile***
  - ***Caching***
  - ***Replication***
- ***Transaction properties of naming operations***

→ ***Naming servers are distributed systems***

CORBA Naming Service

- ***Supports bindings of names to CORBA object references.***
- ***Names are scoped in naming contexts.***
- ***Multiple names can be defined for object references.***
- ***Not all object references need names.***

Naming Contexts



CORBA Names

- *Names are composed of simple names.*
- *Simple names are value-kind pairs.*
- *Value attribute is used for resolving names.*
- *Kind attribute is used to provide information about the role of the object.*

IDL Types for Names

```
module CosNaming {
 typedef string Istring;

 struct NameComponent {
   Istring id;
   Istring kind;
 };
 typedef sequence <NameComponent> Name;
 ...
};
```

The IDL Interfaces

- ***Naming Service is specified by two IDL interfaces:***
  - ***NamingContext defines operations to bind objects to names and resolve name bindings.***
  - ***BindingInterator defines operations to iterate over a set of names defined in a naming context.***

Excerpt of NamingContext Interface

```
interface NamingContext {
    void bind(in Name n, in Object obj)
           raises (NotFound, ...);
    Object resolve(in Name n)
         raises (NotFound,CannotProceed,...);
    void unbind (in Name n)
         raises (NotFound, CannotProceed...);
    NamingContext new_context();
    NamingContext bind_new_context(in Name n)
                  raises (NotFound, ...)
    void list(in unsigned long how_many,
              out BindingList bl,
              out BindingIterator bi);
};
```

Excerpt of BindingIterator Interface

```
interface BindingIterator {
  boolean next_one(out Binding b);
  boolean next_n(in unsigned long how_many,
                 out BindingList bl);
  void destroy();
}
```

Limitations

- *Limitation of Naming: Client always has to identify the server by name.*
- *Inappropriate if client just wants to use a service at a certain quality but does not know from who:*
  - *Automatic cinema ticketing,*
  - *Video on demand,*
  - *Electronic commerce.*