- Name no more precise →
  Interesting/advanced Topics in Operating Systems
  - scalability
  - systems security
  - modeling
- Some overlap with „Distributed Systems“ (Prof Schill) and some classes by Prof Fetzer
- In some cases no written material (except slides)

- Several lectures presented by research-group members.
- Strongly suggested: register for mailing list !!!
  - for questions and discussions
  - only way to inform about short-term issues
  - use: “tu-dresden” mail-adresses

SCALABILITY IN COMPUTER SYSTEMS EXAMPLE: DNS/BIND

HERMANN HÄRTIG, DISTRIBUTED OPERATING SYSTEMS, SS2018

GOAL OF LECTURES ON SCALABILITY

Outline:

- scalability: terminology, problems
- basic approaches
- case studies

Goal:

- understand some of the important principles how to build scalable systems

OUTLINE AND GOAL OF TODAY'S LECTURE

Outline:

- scalability …
- names in Distributed Systems:
  purposes of naming, terminology
- application of scalability approaches on name resolution

Goal:

- understand some of the important principles how to build scalable systems … using DNS as example

MORE CASE STUDIES IN THE CLASS

- memory consistency
- locks and advanced synchronization approaches
- file systems
- load balancing (MosiX) and HPC

GENERAL DEFINITION: SCALABILITY

Scalability:

- the ease with which a system or component can be modified to fit the problem area http://www.sei.cmu.edu/str/indexes/glossary/

Dimensions of Scalability:

- resources: CPUs, memory
- software (versions, better libs, etc.)
- heterogeneity (different hardware / SW = portability)

SPECIFIC: SCALABILITY IN SYSTEMS

- A system is described as scalable if it remains effective when there is a significant increase in the number of resources and the number of users.
(Coulouris, Dollimore, Kindberg: Distributed Systems)

- Scalability [in telecommunication and software engineering] indicates the capability of a system to increase performance under an increased load when resources (typically hardware) are added (Wikipedia)

A SW ENGINEERING ASPECT OF SCALABILITY

Prepare for change in functionality

- software engineering
- choose sufficiently large logical resources
- provide hooks for extension

Not subject of the course

PROBLEMS

- performance bottlenecks / Amdahl's Law
- failures / abuse
- administration

AMDAHL'S LAW

- f: fraction of computation that can be enhanced

- Speedup: $\frac{\text{original execution time}}{\text{enhanced execution time}}$

- S: speedup factor for f

- $\text{Speedup}(f,S) = \frac{1}{\left(1-f+\frac{f}{S}\right)}$

AMDAHL'S LAW FOR PARALLEL COMPUTING

- attack the common case
- if S becomes VERY large, speedup approaches $\frac{1}{(1-f)}$

interpretation for parallel systems:

- P: section that can be parallelized
- 1-P: serial section
- N: number of CPUs
- $\text{Speedup}(P,N) = \frac{1}{\left(1-P+\frac{P}{N}\right)}$

THE “RPC” PRINCIPLES

- partitioning
  - split systems into parts that can operate independently to a large extent
- replication
  - provide several copies of components
    - that are kept consistent eventually
    - that can be used in case of failure of copies
- locality (caching)
  - maintain a copy of information that is nearer, cheaper/faster to access than the original

MORE PRINCIPLES FOR SCALABILITY

- identify and address bottlenecks (!!!)

- specialize functionality/interfaces
- right level of consistency
  caches, replicates, … need not always be fully consistent
- lazy information dissemination
- balance load

MORE ON LOAD BALANCE

- balance load
  - keep load under reasonable threshold
    - at the processing components
    - in the communication subsystems
- load balancing can be static or dynamic.
- Will study a detailed example for dynamic load balancing later(MosiX).

SOME MORE ISSUES

- minimize the delay induced by “RPC”
- prepare for change
- information dissemination

DOMAIN NAME SYSTEM AS AN EXAMPLE

- names and name resolution etc in general
- a bit of history of internet names
- DNS general properties
- RPC in DNS

NAMES, IDENTIFIERS, ADDRESSES

- names
  - symbolic
  - have a meaning for people
- identifiers
  - identifies a component (uniquely)
  - are used by programs
- addresses
  - locates a component & can change
  - can change

NAME RESOLUTION

- name resolution:
  - map symbolic names to objects
  - indetails: to a set of attributes such as:
    identifiers, addresses, other names, security properties

- Principle interface:
  - Register (Name, attributes, ...)
  - Lookup (Name) -> attributes

RELATED

- compilers
  - statically map names to addresses
- dynamic libraries
  - dynamically remap addresses
- port mapper (SUN RPC)
  - map service to port
- Name resolution is a form of dynamic mapping of pathnames to attributes.

OBSERVATION

- Many services, tools, ... provide their own name resolution
  - file systems (UNIX: path names to I-Nodes)
  - login
  - RPC (remote procedure call) systems (portmapper)

PURPOSE OF DIRECTORY SERVICES

- integration of name services
- generic name service
- world-wide use of names
- pervasively used:
  - email/web
  - computer attributes (IP addresses)
  - people attributes (certificates, …)
  - …

A BIT OF HISTORY

- UUCP/MMDF (cum grano salis):
  - ira!gmdzi!oldenburg!heinrich!user (path to destination)
  - user@ira!heinrich%gmdzi   (mixing identifiers and path information)

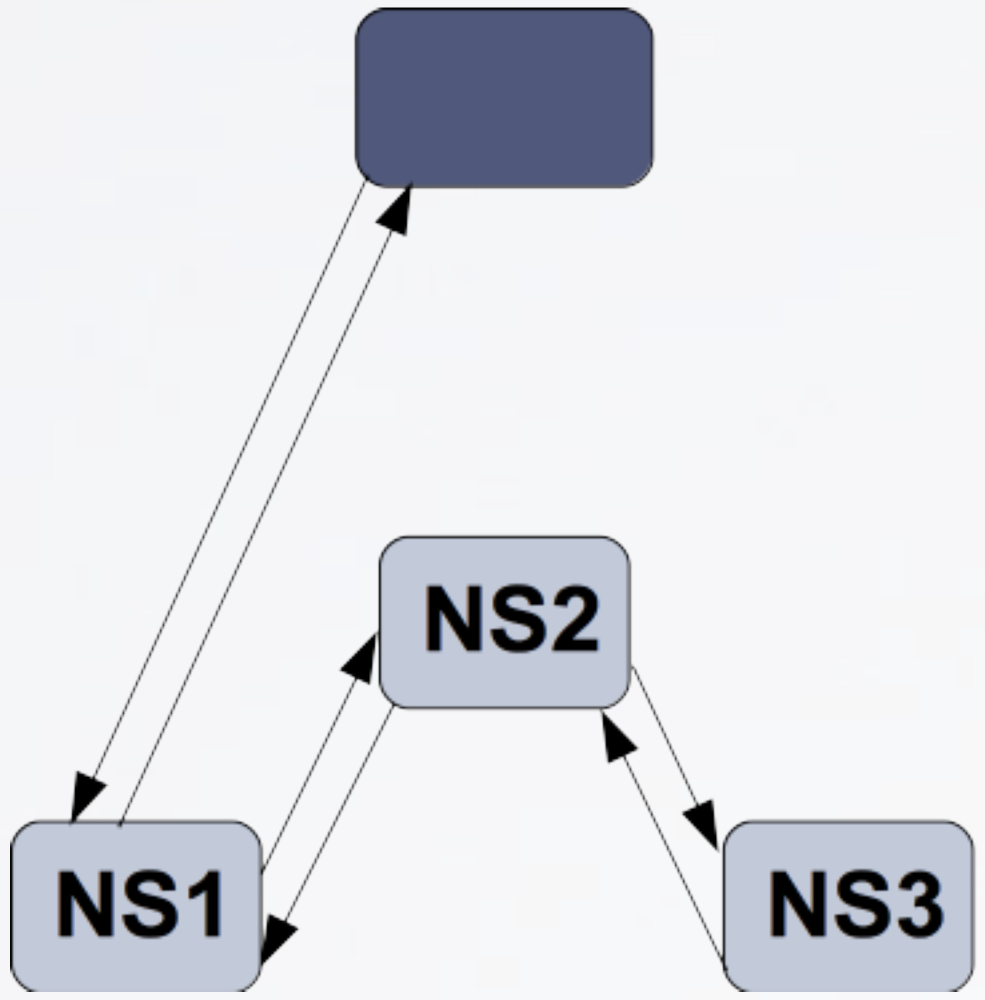
A BIT OF HISTORY

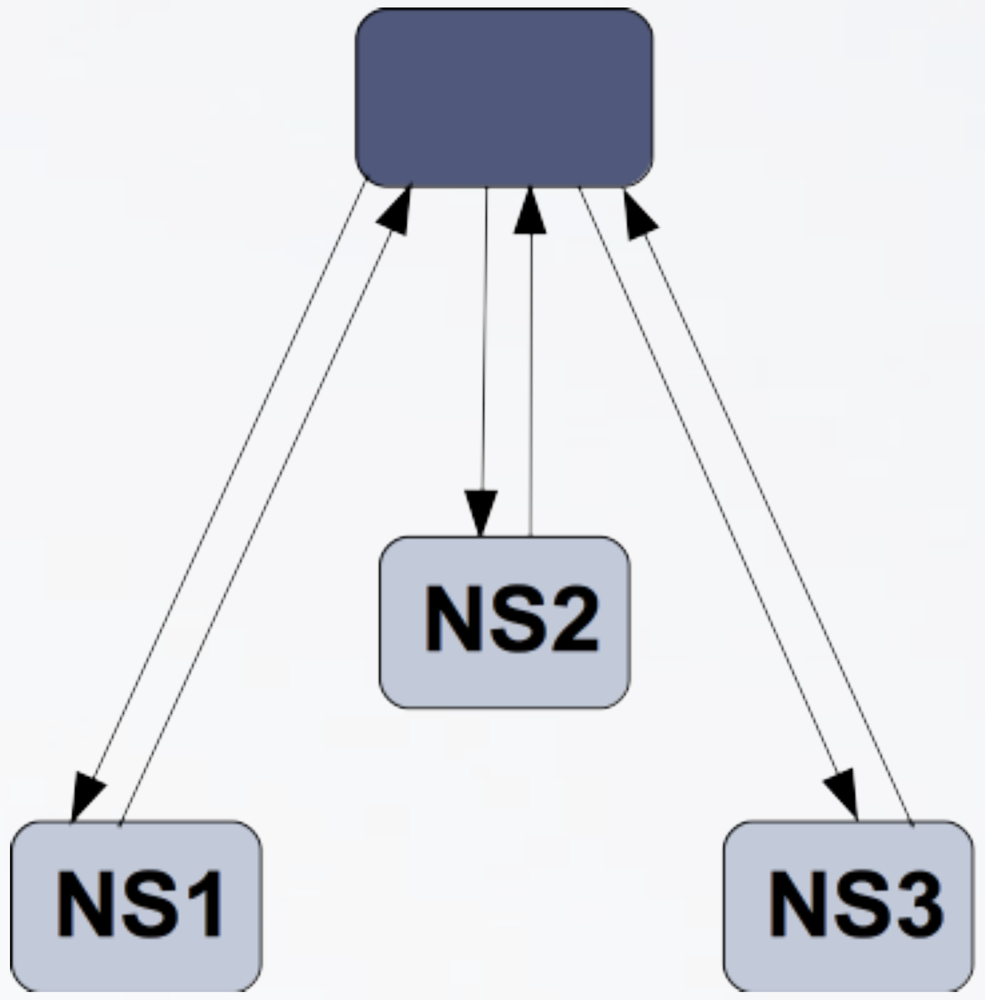
- ARPA-Net at the beginning:
  - a single file: hosts.txt
  - maintained at Network Information Center of SRI (Stanford)
  - accessed via ftp
  - TCP/IP in BSD Unix => chaos name collisions, consistency, load
- DNS: Paul Mockapetries (84) ...

SOME DNS TERMINOLOGY

- naming domain
  subtree in the hierarchy of DNS contexts
- zone
  (aka Zone of authority) Subset of a domain over which an authority has complete control. Subzones (starting at apices of a zone) can be delegated to other authorities.
- navigation
  querying in a set of cooperating name spaces

RECURSIVE ./. ITERATIVE

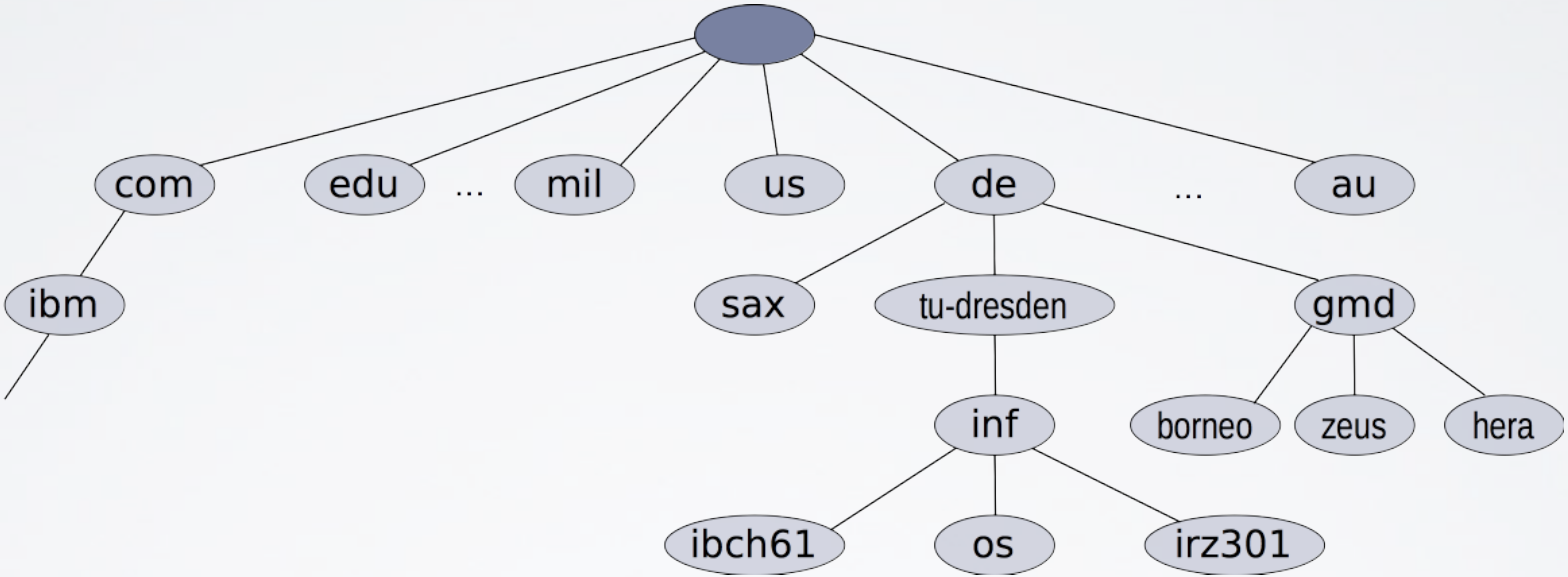




REQUIREMENTS / PROPERTIES

- arbitrarily large numbers
- arbitrary units of administration
- long living names, the higher in the hierarchy the longer
- high robustness
- restructuring of name spaces
- consistency
- efficiency

DNS NAME SPACE (ORIGINAL)

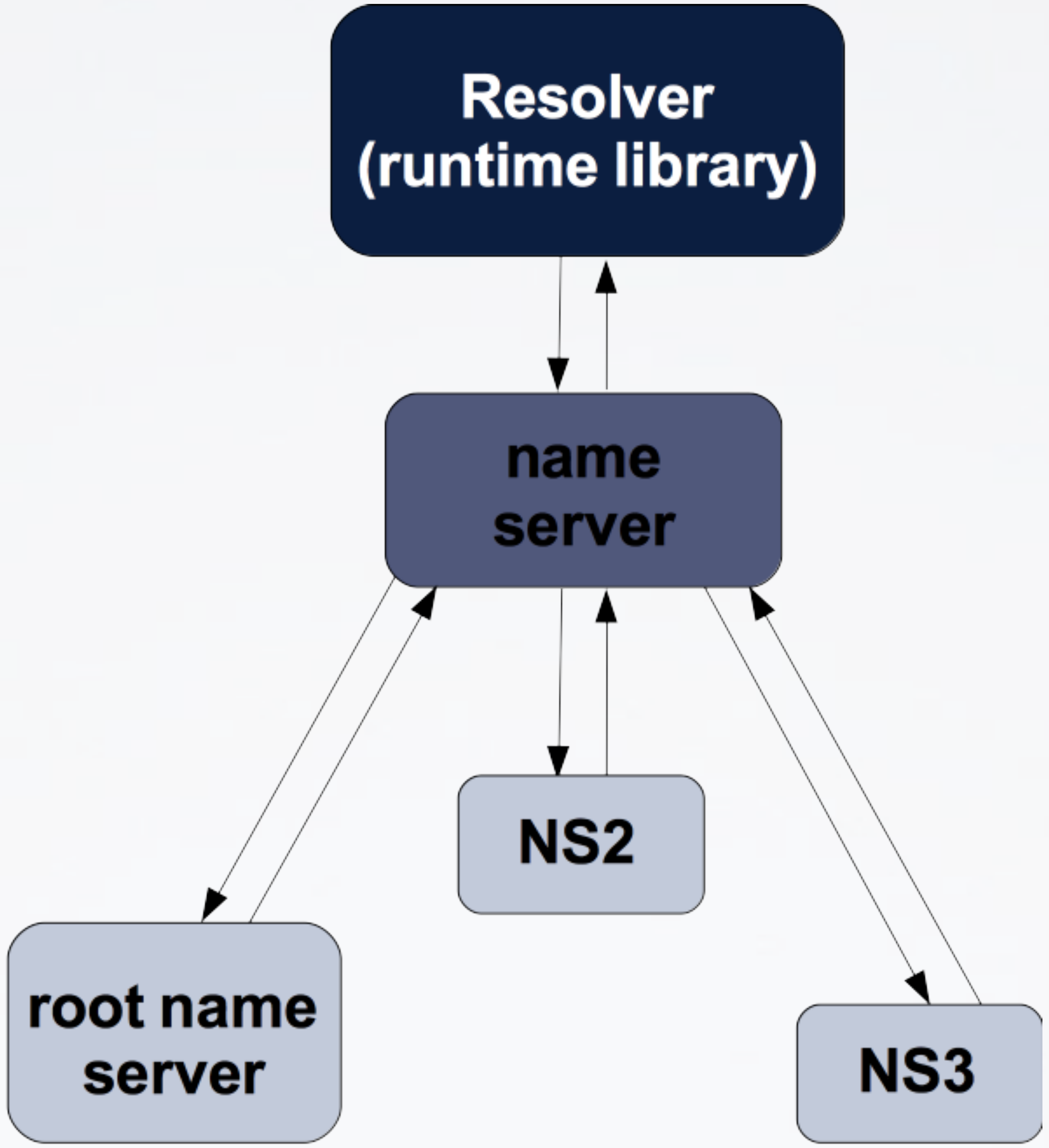


TODAY: hundreds of “top level domains"

EXAMPLES

- inf.tu-dresden.de — domain
- os.inf.tu-dresden.de — computer
- heidelberg.ibm.com — domain
- ftp ftp.inf.tu-dresden.de
  - DNS: → IP address: 141.76.2.3
  - ftp daemon: IP address, port 21
- properties:
  location independent / not very deeply nested

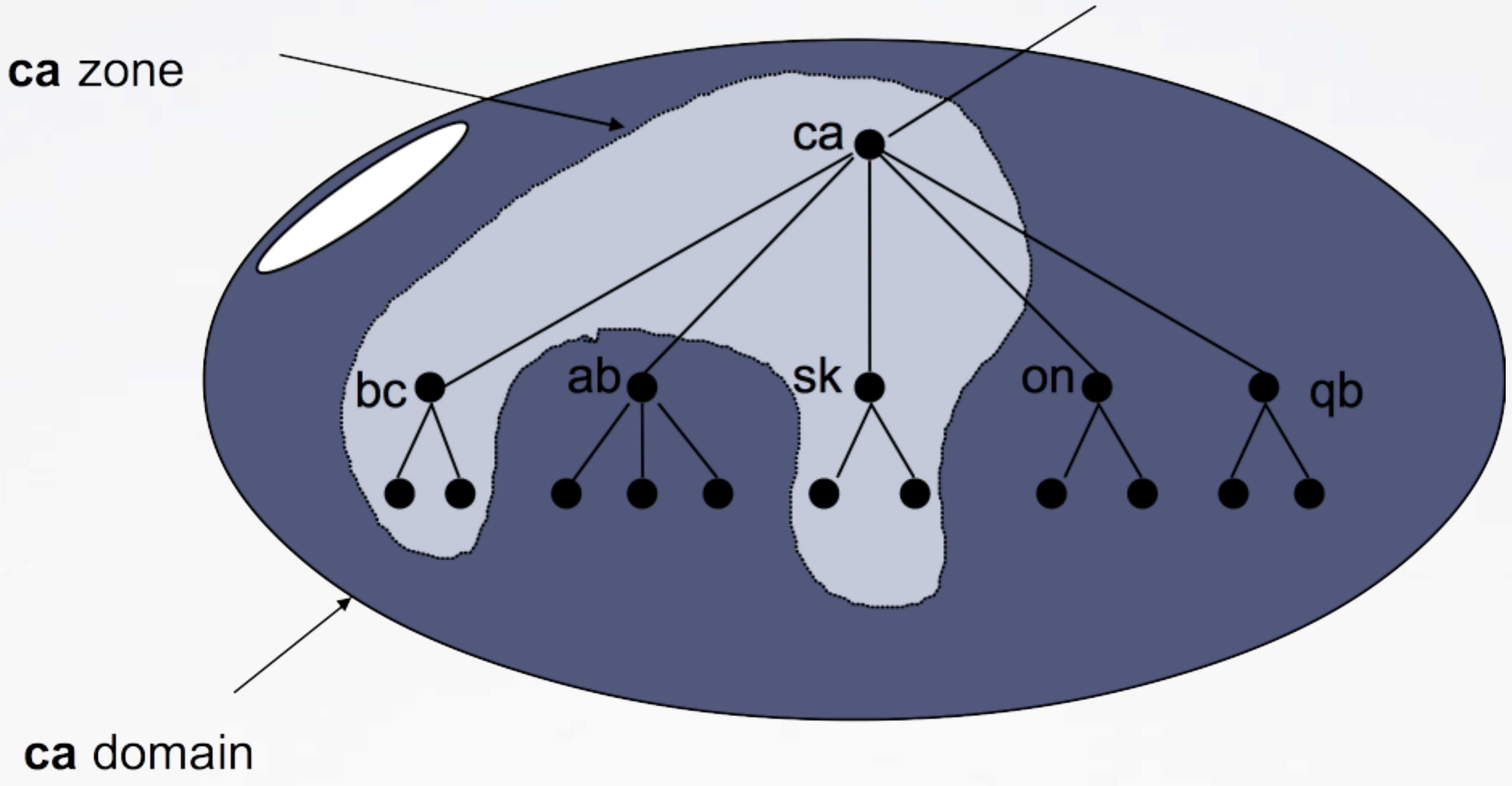
IMPLEMENTATION STRUCTURE (BIND)



RPC: PARTITIONS

- Zones:
  - administrative unit
- Name Server:
  - maps to names and addresses of name servers responsible for sub zones
  - maintains management data
  - process doing the name resolution for one zone
- key interface: Resource records (RR)

ZONES



example taken from Coulouris et al, Distributed Systems

RPC: REPLICATION

- 2 ways of replication:
  - several IPs/names
  - “any cast" (send packet to one of many servers with same IP)
- 13 root name server IPs, several hundreds of any cast
- each zone has at least
  one primary and one secondary IP

RPC: CACHING

- each name server caches resource records
- time to live attribute
- authoritative versus non-authoritative answers

RESOURCE RECORDS

Record type	Interpretation	Content
A	address	IPv4 address
AAAA	address	IPv6 address
NS	Name server	DNS name
CNAME	Symbolic link	DNS name of canonicial name
SOA	Start of authority	Zone-specific properties
PTR	IP reverse pointer	DNS name
HINFO	Host info	Text description of host OS
...	...	...

LITERATURE

- Paul Albitz & Cricket Liu
  DNS and BIND
  O´Reilly & Associates, Inc.
- Mark Hill, Michael Marty
  Amdahl's Law in the Multicore Era IEEE
- Couluris, Tollimore, Kindberg
  Distributed systems