Welcome to

CS131

Parallel and Distributed Systems

Prof. A. Veidenbaum

Today's topics

- Distributed systems design goals
- Types of distributed systems

- Reading assignments:
  - Today: Ch. 1, Sec. 1.2
  - Next time: Ch. 1, Sec. 1.3 - 1.3

  - Plz complete reading assignments ***before*** next class

Summary of last lecture

- **A distributed system is a collection of independent computers that appears to its users as a single coherent system.**

- **Middleware sftw is what makes it a DS**
  - **Somewhat like an OS for a single computer**
  - **Middleware provides**
    - » **Resource management**
    - » **Facilities for inter-application communication.**
    - » **Access management**
    - » **Security services.**
    - » **Masking of and recovery from failures.**

Major DS Design Goals

- Allow users to easily connect to resources
- Transparency
  - hide the fact that resources are distributed
- Openness
  - allow extendibility, interoperability
- Scalability
  - An ability to increase system size without problems
- Security, Fault-tolerance, etc

Why Connect Users/Resources

- **Distributed systems allow resource sharing**
  - printers, cycle servers, files, documents, storage devices, Web pages, networks, etc
- **Makes sense on many levels**
  - economic reasons
  - ease of information exchange and sharing
  - collaboration
- **Creates new problems**
  - Access management, consistency
  - Security
  - Fault tolerance
  - Fair and controlled resource sharing

Transparency

- **Makes a DS look like a single computer**
- **Users do not need to know where the resources are**
  - **Or that they are shared**
  - **Or that they fail**
  - **Or that they migrate or are replaced**
- **How is it achieved?**
  - **Through multiple layers of software!**

Transparency (Cont'd)

- **Can be applied at multiple levels in a DS**
  - **Access**
  - **Location**
  - **Relocation**
  - **Migration**
  - **Replication**
  - **Concurrency (of use)**

- **Is complete transparency always desired?**
  - **Hint: think of performance**

Openness

- Allows components from different vendors
- Allows new components to be added
- Allows old components to be replaced
- Implies interoperability and portability
- Uses an Interface Definition Language (IDL)
  - specifies syntax for a process to talk to another process which provides a given interface
    - » E.g. function names, parameters, etc
  - NOT the implementation (policy vs mechanism)

Openness (Cont'd)

- Use of an IDL allows
  - Different implementations of a given interface
  - Many to one mapping for a service
- A DS needs an ability to change
  - For instance, change/modify/improve a component
- Separating policy from mechanism allows this
  - Book example – a Web cache with a number of policies
    - » Where is the data cached?
    - » What is the replacement policy?
    - » Is it shared or private?
    - » Consistency with the source

Scalability

- An ability to grow the system
  - One of the key advantages of distributed systems
    - » Adapt to new requirements, such as more users
- Centralization of resources limits scalability
- Replication and Caching improves scalability
  - BUT they lead to *consistency* problems
    - » Multiple copies exist *and* may be different
      - Have you encountered this problem?

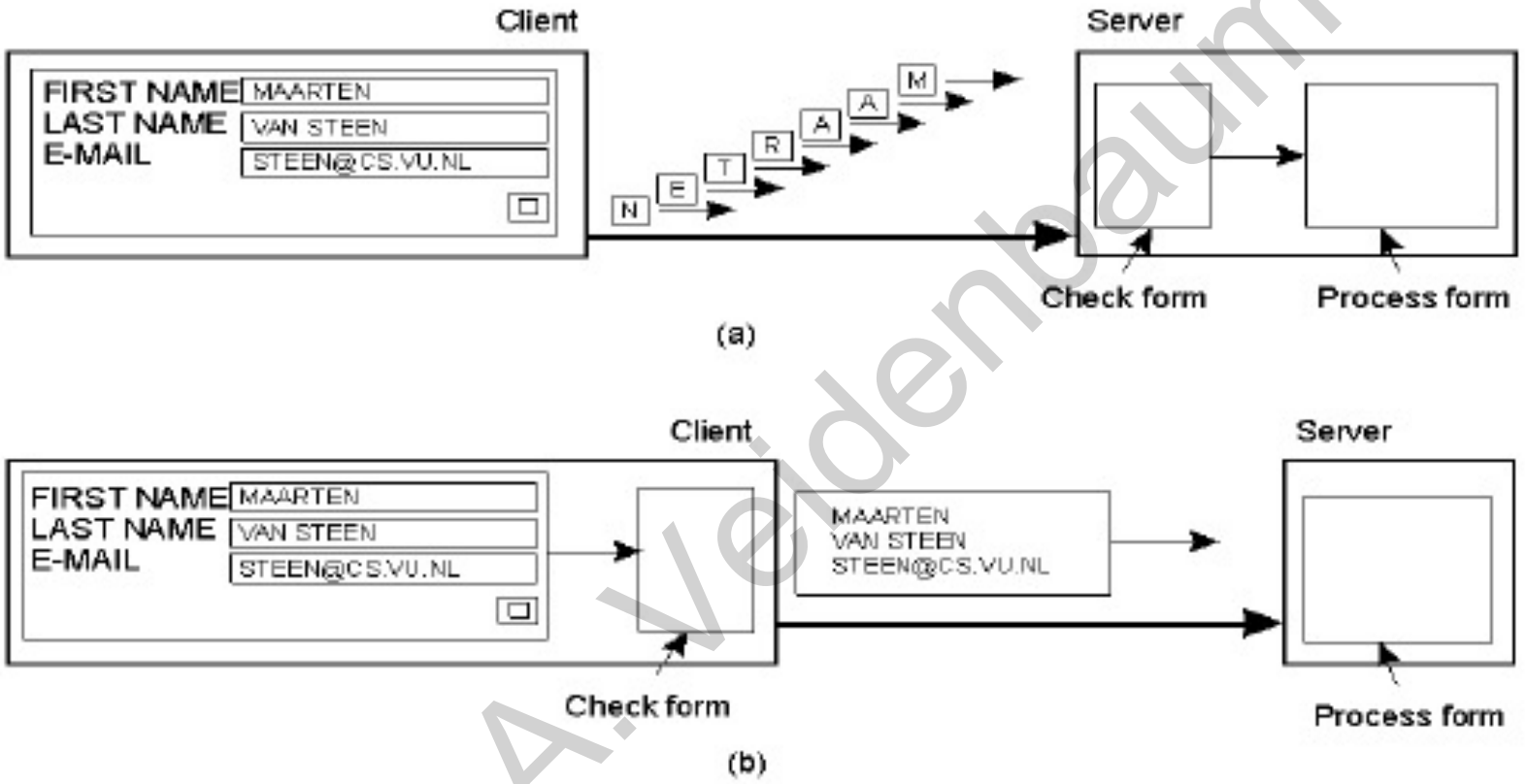
Scalability

- **Can be measured in 3 different dimensions**
  - **Size scalability**
    - » **can more users/resources be added**
  - **geographical scalability**
    - » **with respect to the distance**
  - **administrative scalability**
    - » **across multiple administrative domains**

Examples of Scalability Problems

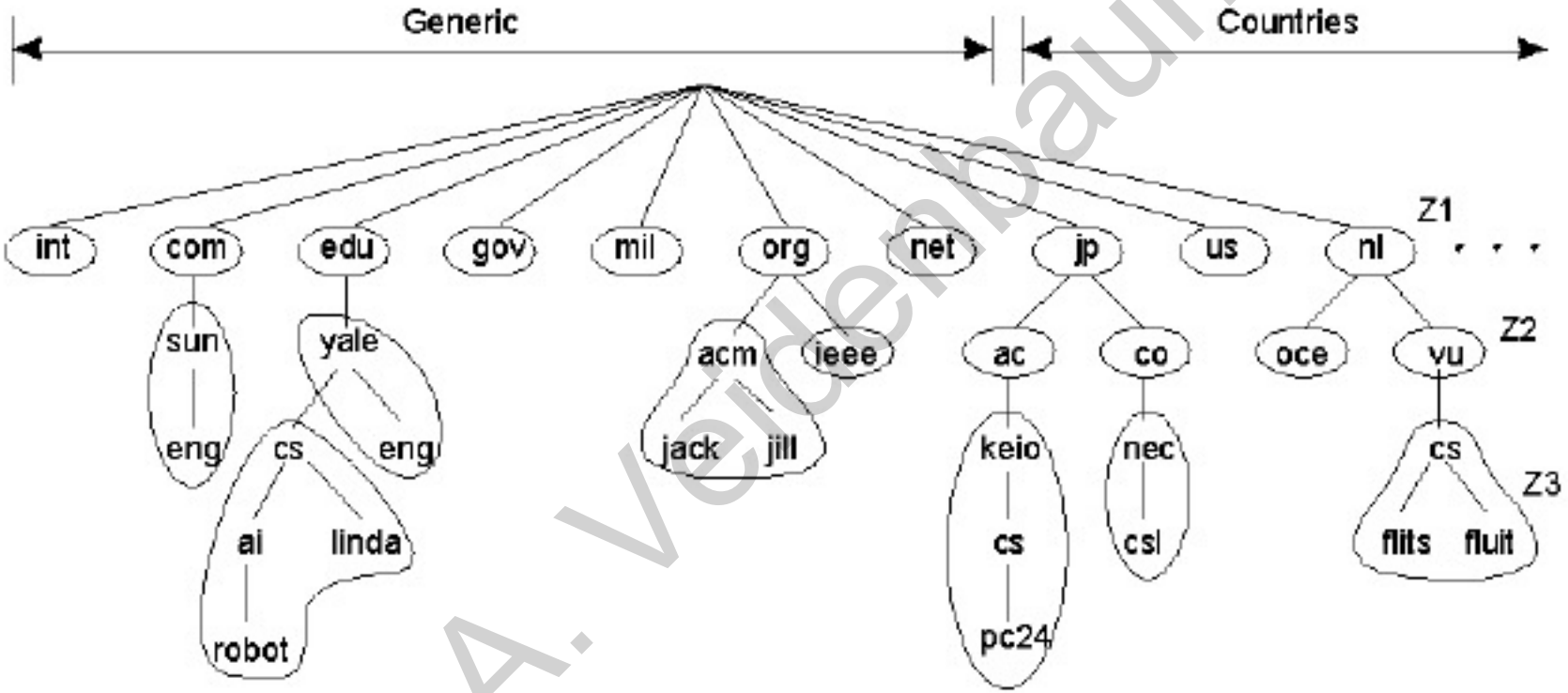
• Concept	• Example
• **Centralized services**	• **A single server for all users**
• **Centralized data**	• **A single on-line telephone book**
• **Centralized algorithms**	• **Doing routing based on complete information**

Scaling Techniques (1) - hiding latency



- Who does the work of checking forms as they are being filled?
- a) is not efficiently using the network and the server

Scaling Techniques (2) – partitioning, distribution



Distribution of service: dividing DNS name space into zones.

Scaling Techniques (3) – replication, caching

- **Replication helps limit performance problems**
  - **Local servers**
  - **Multiple resources - Google data centers**
- **Caching is a special form of replication**
  - **Making a copy of data**
- **Replication/caching lead to consistency problems**
  - **Multiple copies may differ**
    - » **Think of browser (local) caching**

Possible implementations

Consider a large database with many users. How to best implement it?

1. **One central database**
   - **What will this lead to?**

2. **Multiple database partitions distributed somehow**
   - **Better?**

3. **Each computer has a full database**
   - **Problems?**

4. **Replicate partially and Cache locally**

Pitfalls in designing DSs

- **Basically, dubios simplifying assumptions made during the DS design**
  - **The network is reliable**
  - **The network is secure**
  - **The network is homogeneous**
  - **The topology does not change**
  - **Latency is zero**
  - **Bandwidth is infinite**

- **Such assumptions need to be carefully considered!**