The Wizardry of Scaling

Ayende Rahien / Oren Eini

http://ayende.com/Blog

ayende@ayende.com

Scaling

A service is said to be scalable if when we increase the resources in a system, it results in increased performance in a manner proportional to resources added

Werner Vogels
CTO - Amazon.com

Scaling

- # of requests
- Data size
- Complexity

Scenario

Sign in

Username:

Password:

Forgot your password?

Remember me on this computer (?)

Remember my password (?)

Sign in

Use enhanced security

Attempt #1

```
CREATE PROCEDURE CreateUser
                 @username NVARCHAR(50),
                 @email    NVARCHAR(50),
                 @pass     NVARCHAR(50)
AS
  INSERT INTO users
              (username,
               email,
               password)
  VALUES      (@username,
               @email,
               @pass)
```

```
CREATE PROCEDURE LoginUser
                 @username NVARCHAR(50),
                 @pass     NVARCHAR(50)
AS
   SELECT 1
   FROM   users
   WHERE  [User] = @username
          AND password = @pass
```

Users like us!

More Users

Get Bigger Server

Adding external API

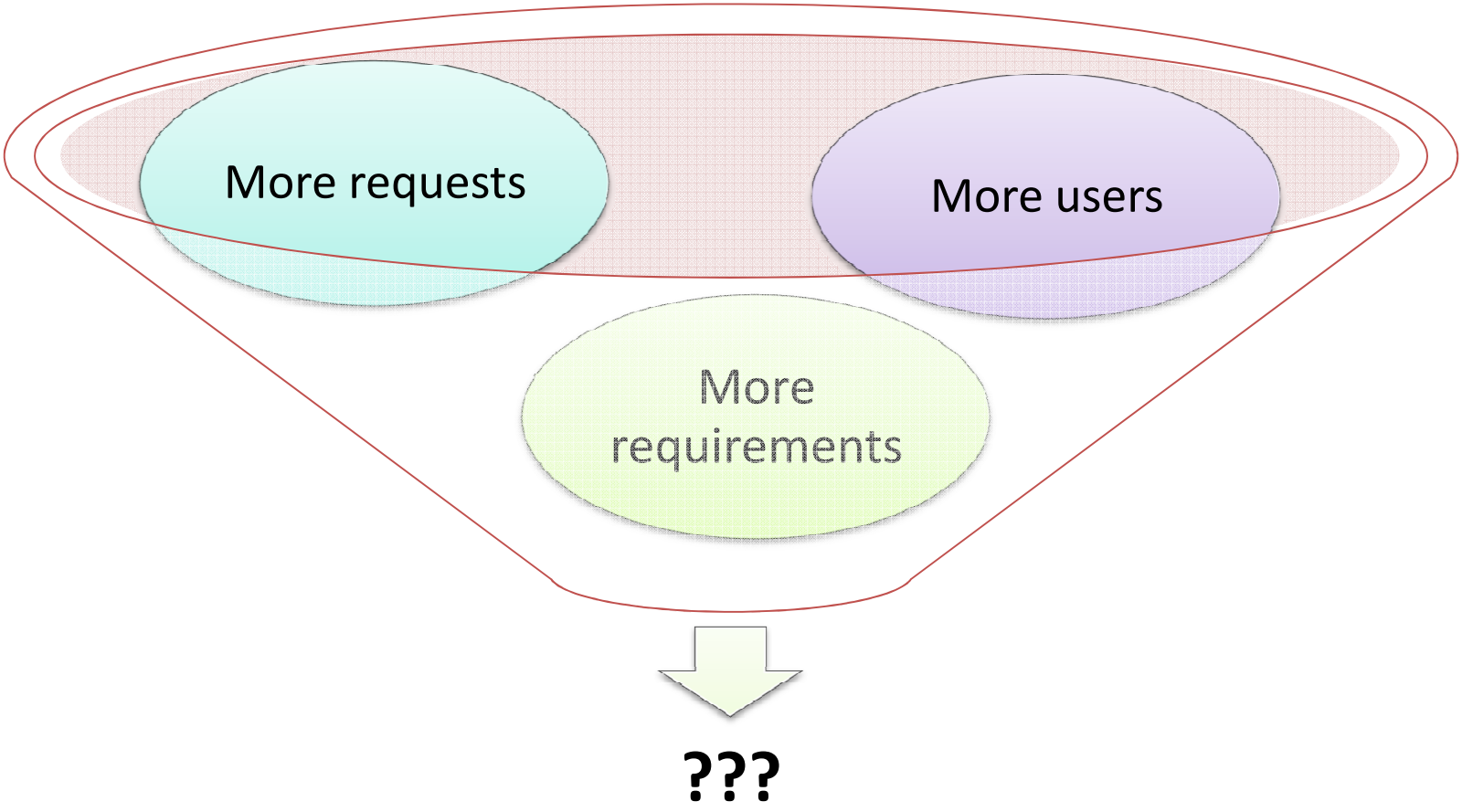
More auth requests

Get Bigger Servers

Requirements

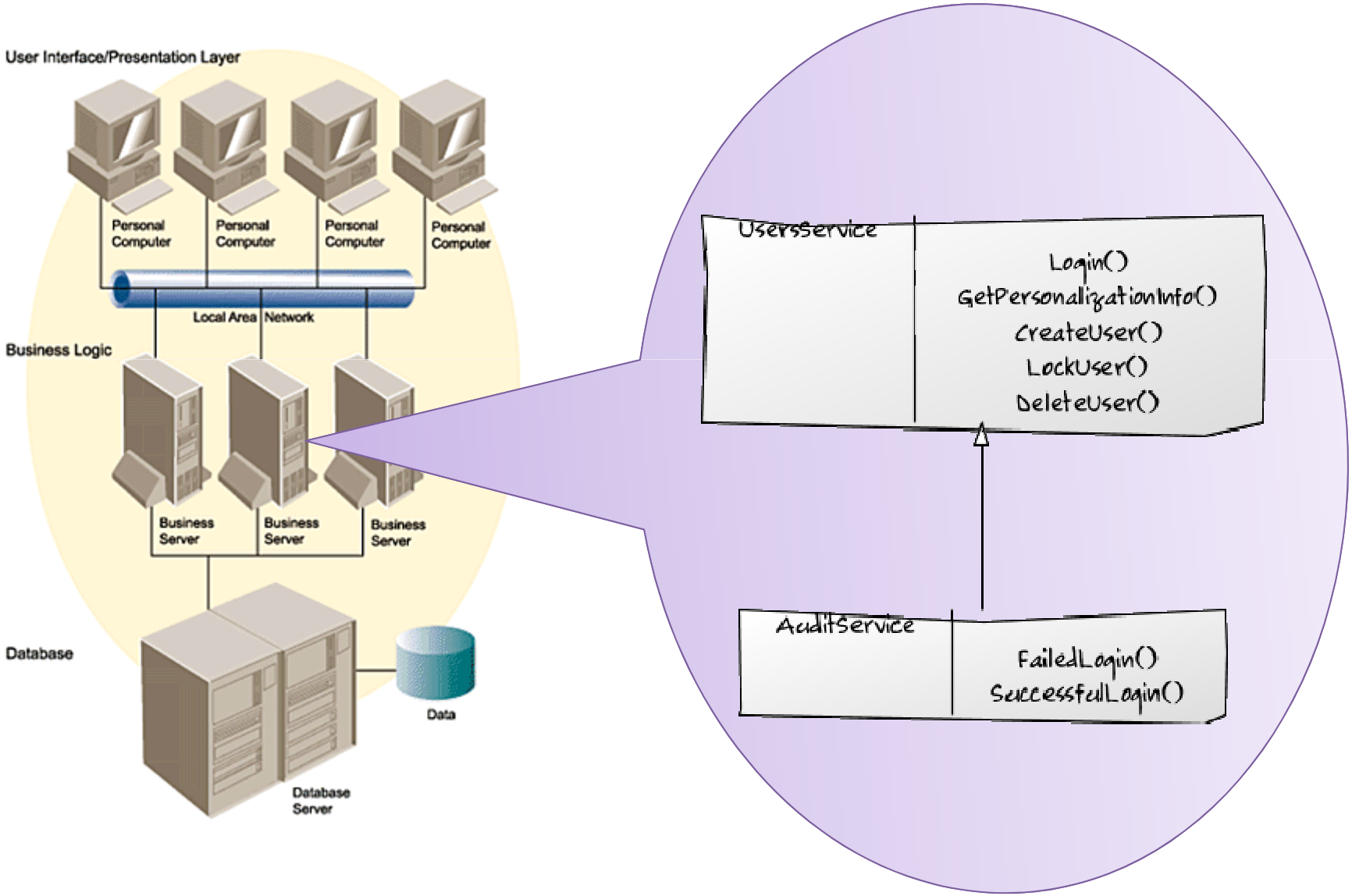
- Lock accounts
- Personalization
- Auditing

End Result



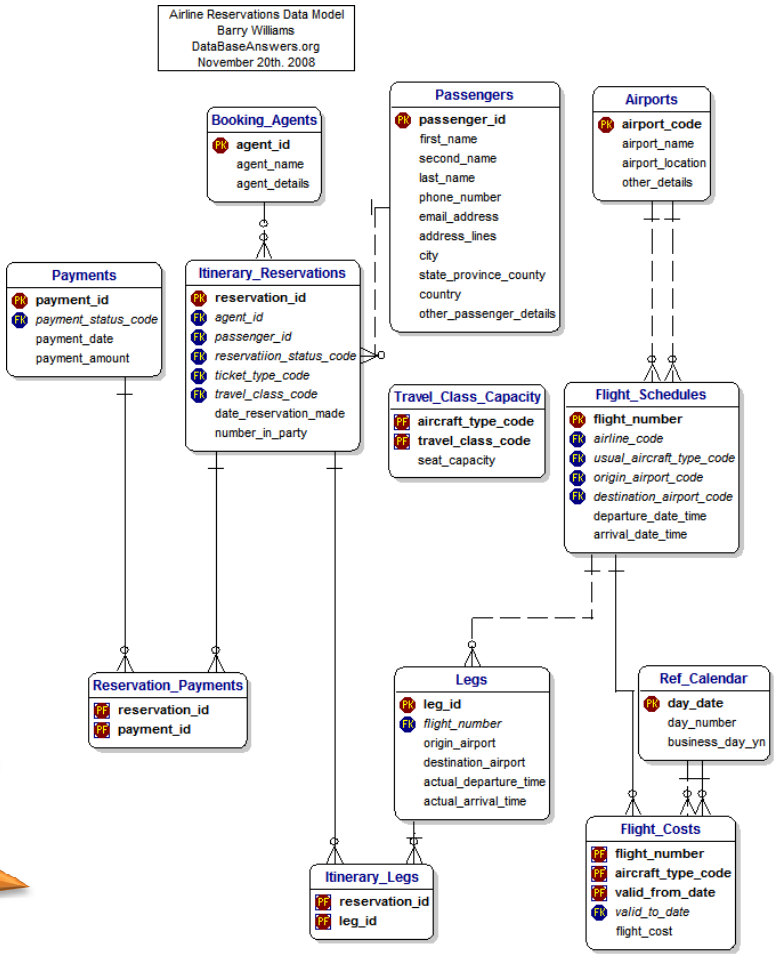
Micro optimization as a way of life

Traditional Architecture



A single source of truth?

Audit



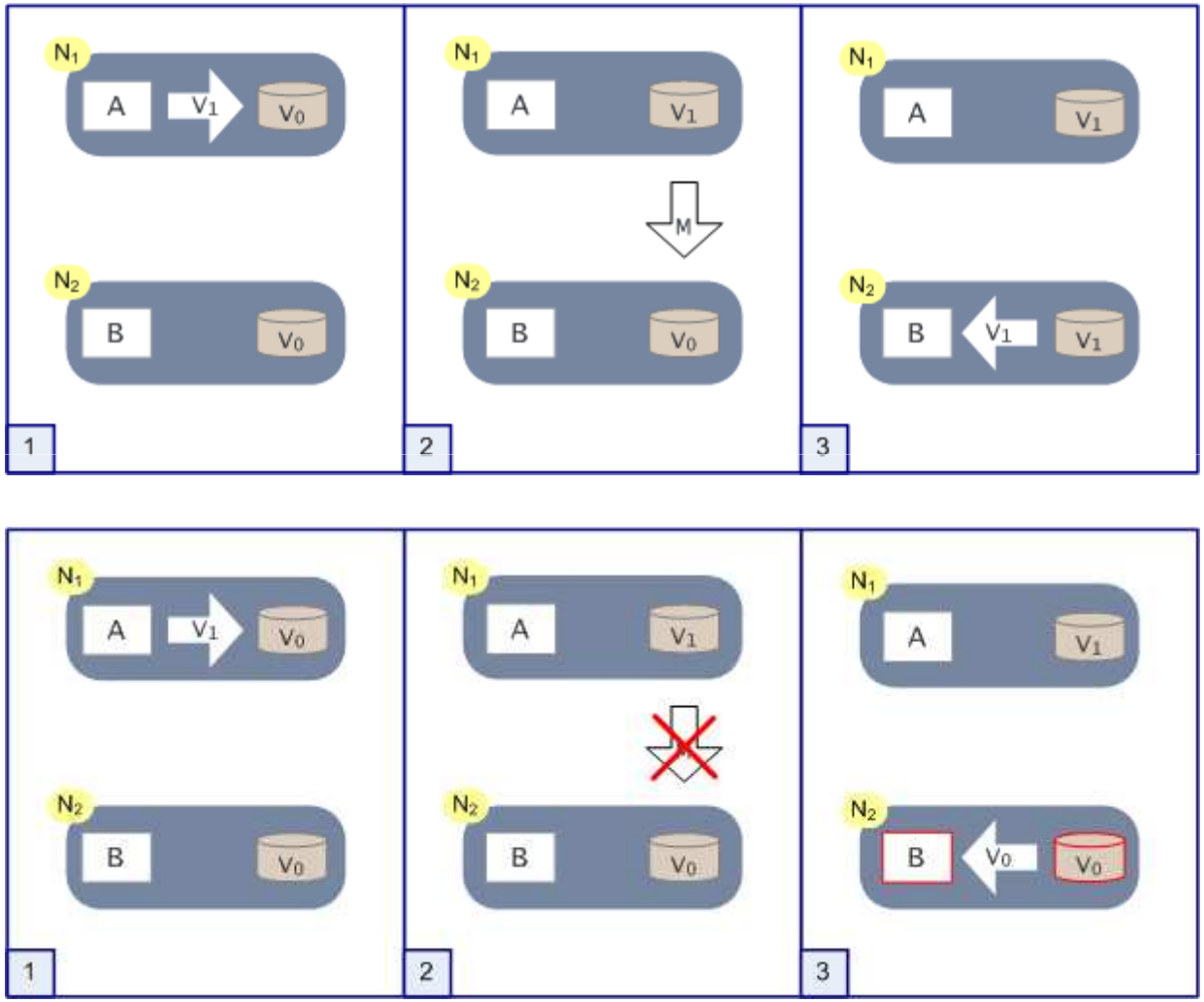
Login

OLTP

CAP

- Consistency
- Availability
- Partition Tolerance

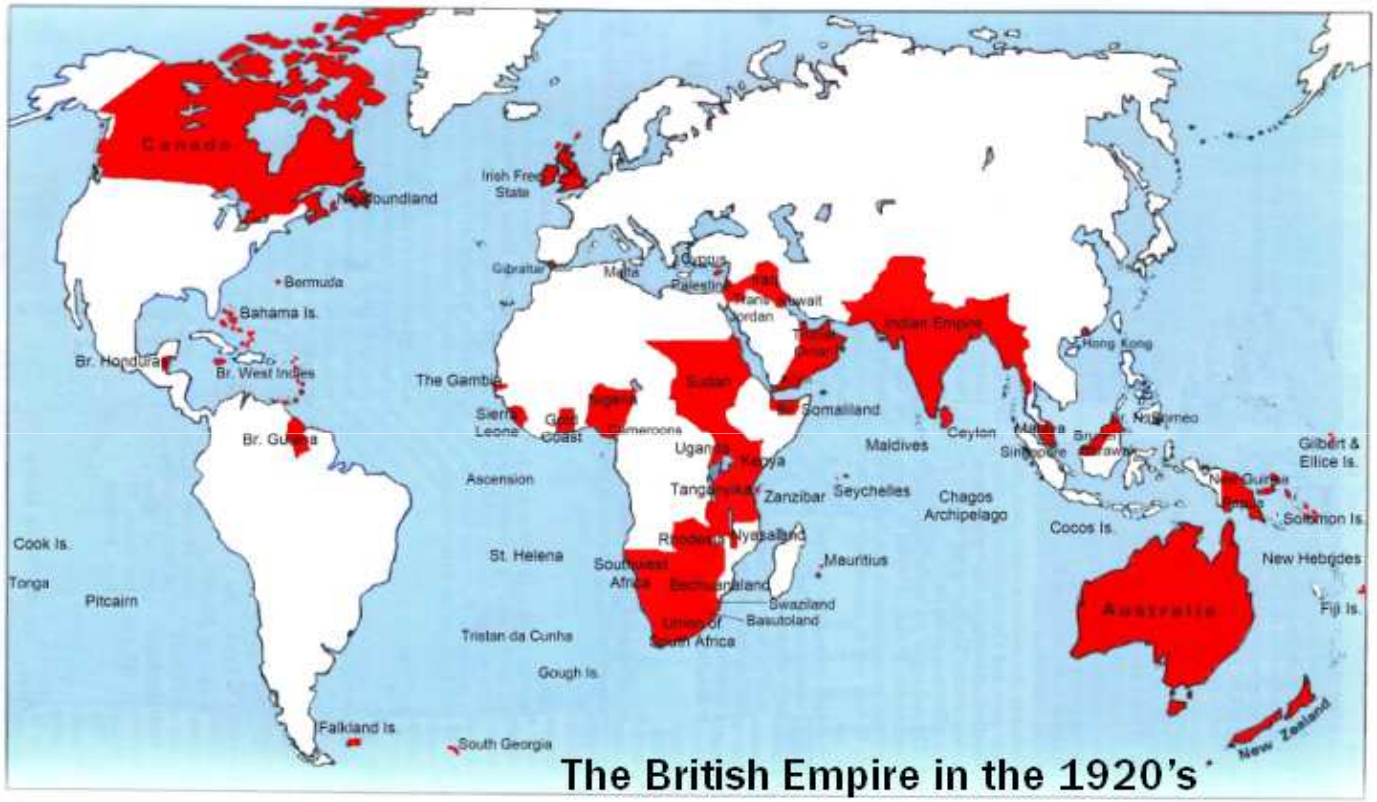
CAP



http://www.julianbrowne.com/article/viewer/brewers-cap-theorem

How to scale my app?

Divide and Conquer

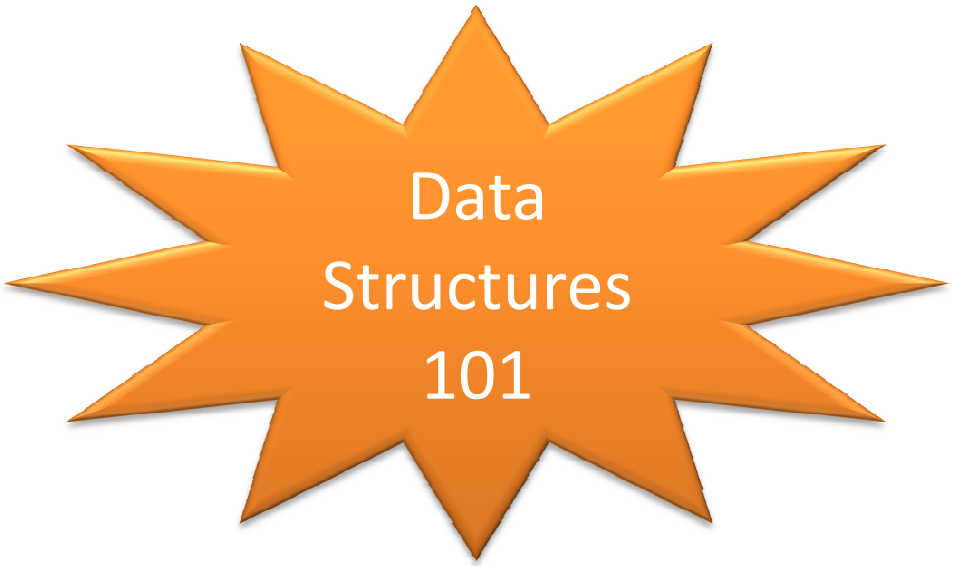


Operations

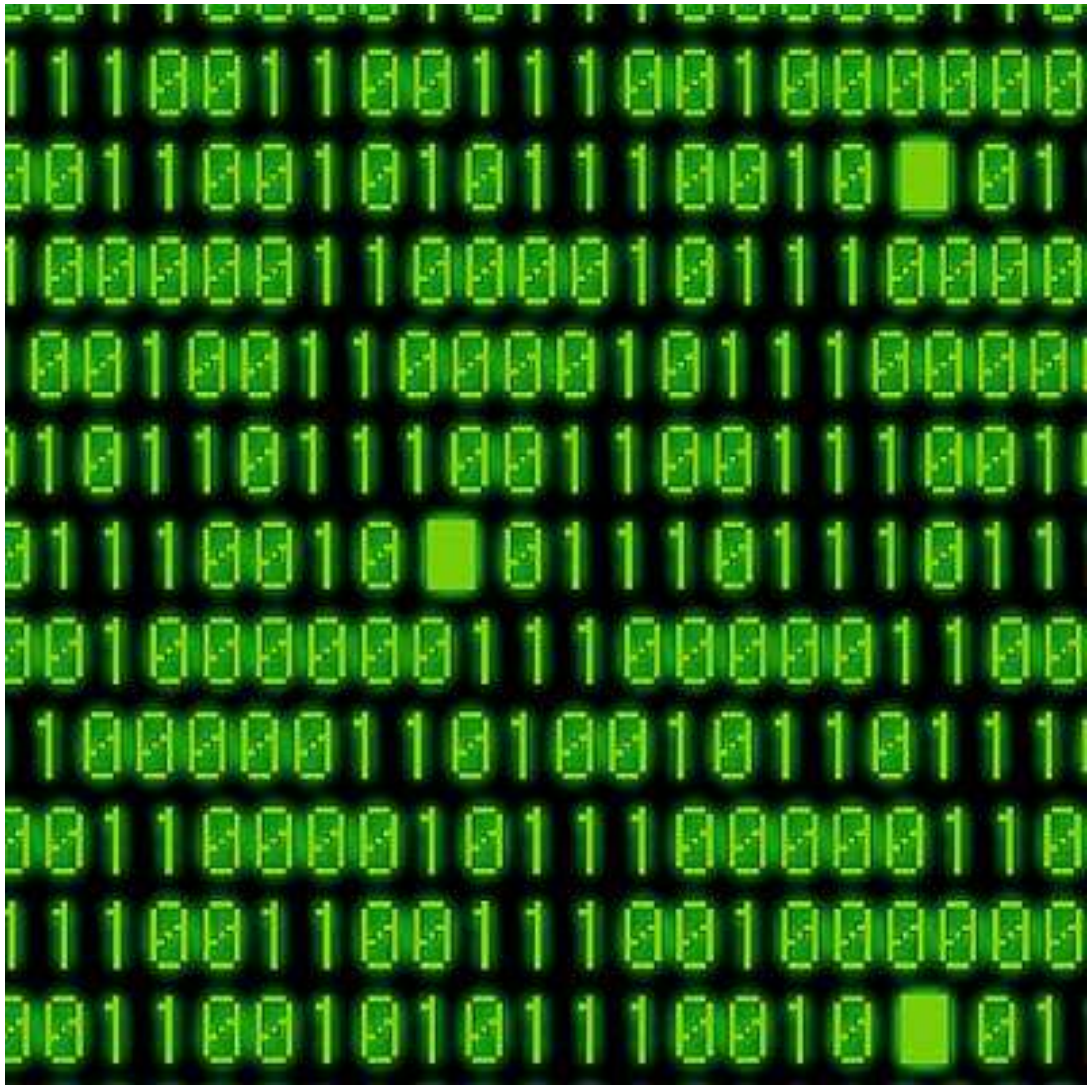
- **Authenticate User**
- Audit all logins
- Create User
- Personalization information

Solve just a single problem

For N number of users, answer if user/pass combo is valid in constant time



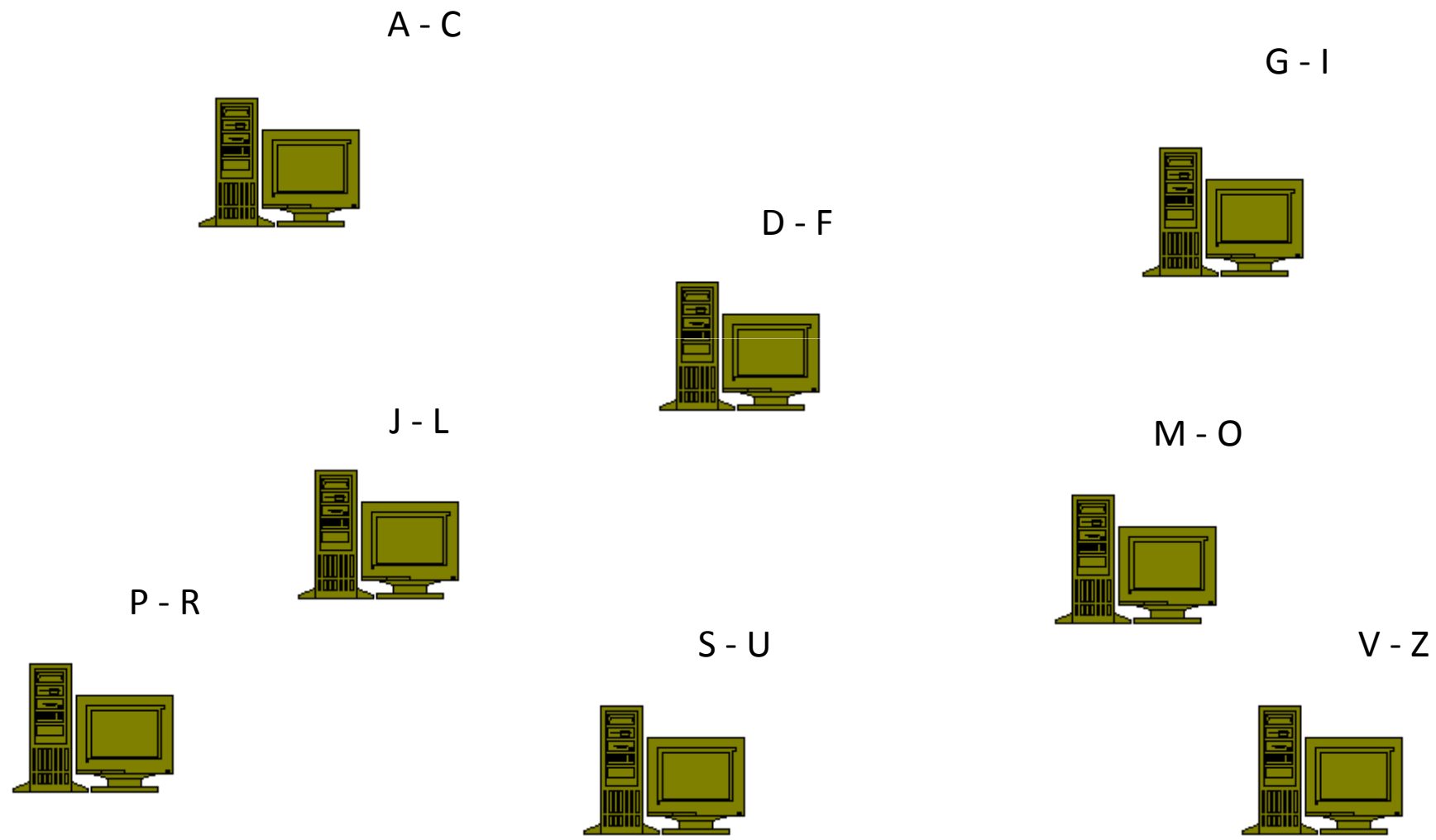
Code



In memory / single machine

- Single user memory usage – 56 bytes:
  - 16 chars / 32 bytes – username
  - 24 bytes – hashed password
- 1,000,000 users = 53.4 Mb!
- 10,000,000 users = 534 Mb!
- 100,000,000 users = 5.34 Gb!

Scaling higher…



Operations

- Authenticate User
- **Audit all logins**
- Create User
- Personalization information

Audits

Record information about login attempts results

System.IO.File

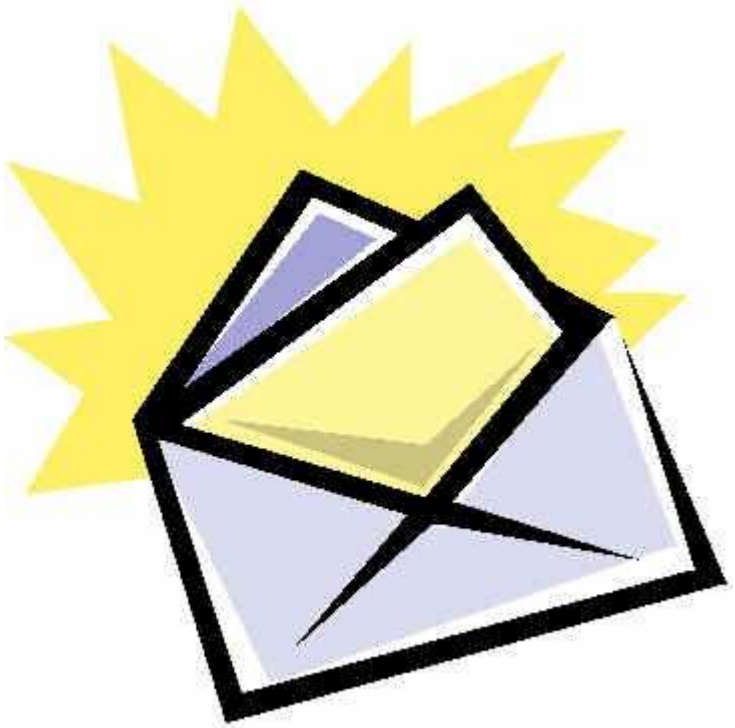
Write behind

Operations

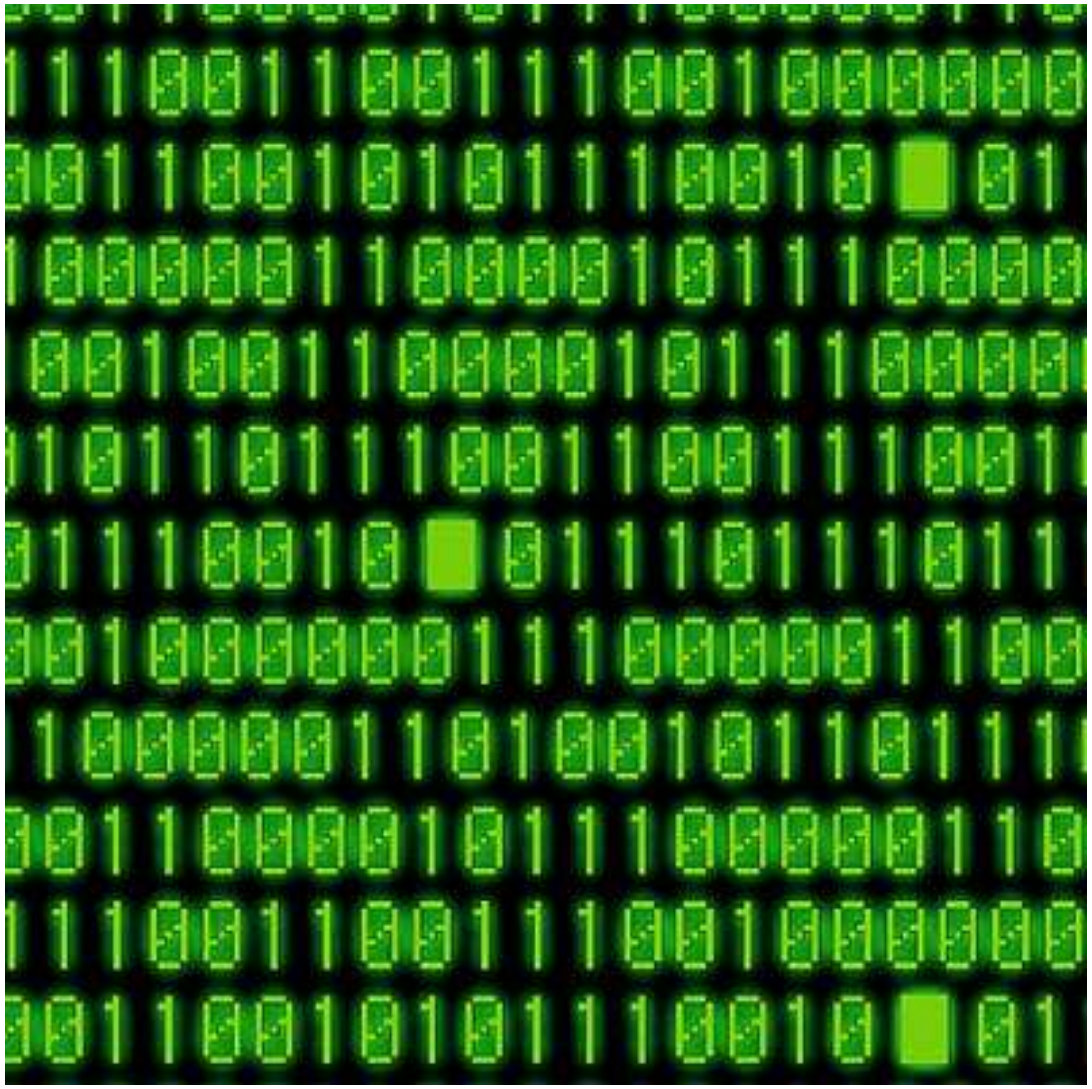
- Authenticate User
- Audit all logins
- **Create User**
- Personalization information

Create User

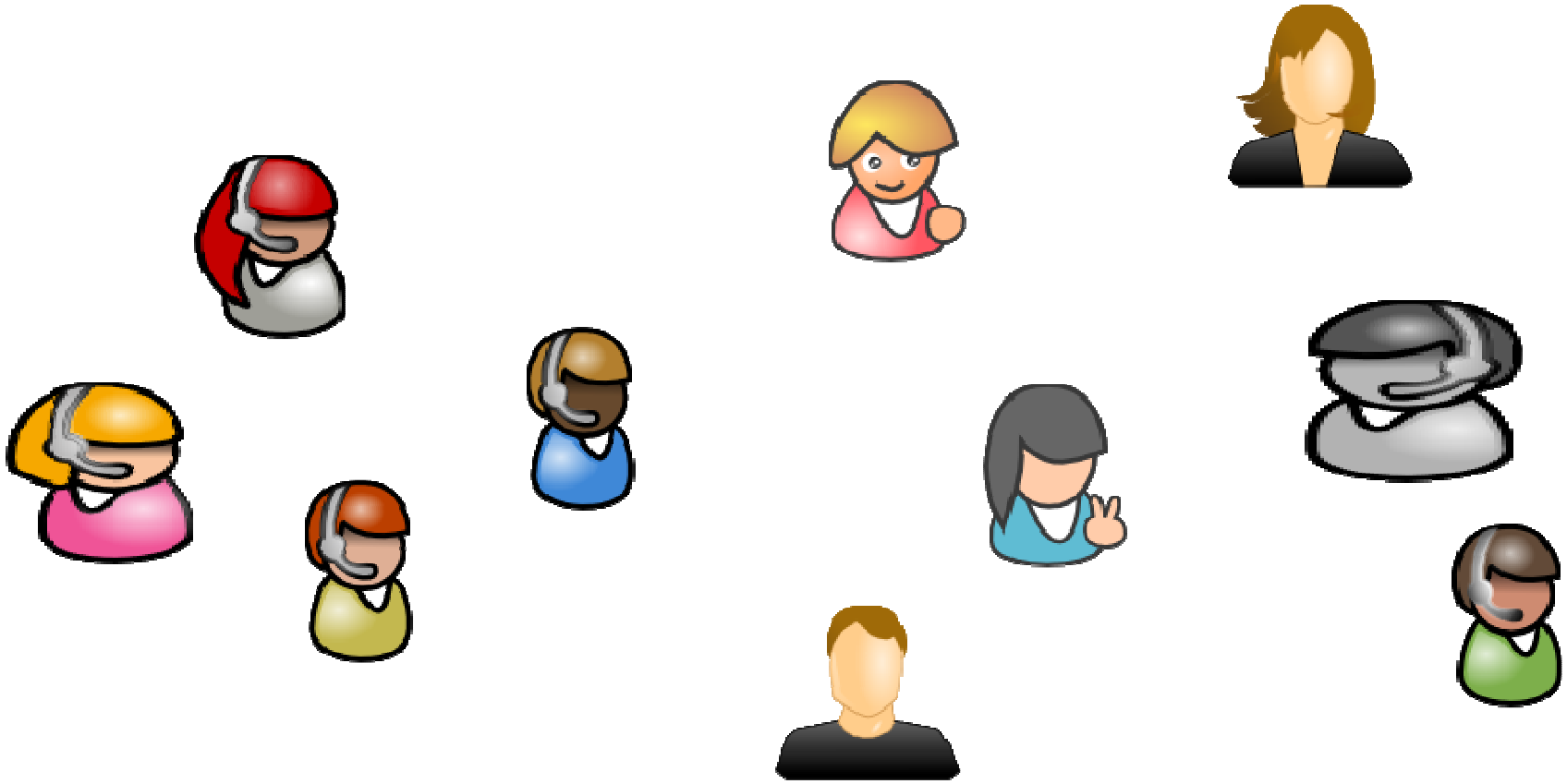
- Email cycle
- Register in system:
  - Personalization
  - Login
  - Etc…



Code



A system of idiot savants

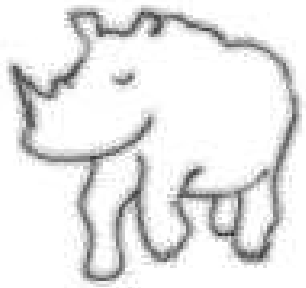


Tailoring the solution

- When breaking a system to independent components…
  - 3 tiers doesn't make sense
  - Sharing among components is discouraged
  - Find the best solution for the problem *at hand*, vs. generic solutions.

In .NET – Service Buses

In .NET – Non SQL DBs



Summary

- Divide and conquer
- Single task operations
- Optimize for each task
- There is no one infrastructure fits every task

Questions?