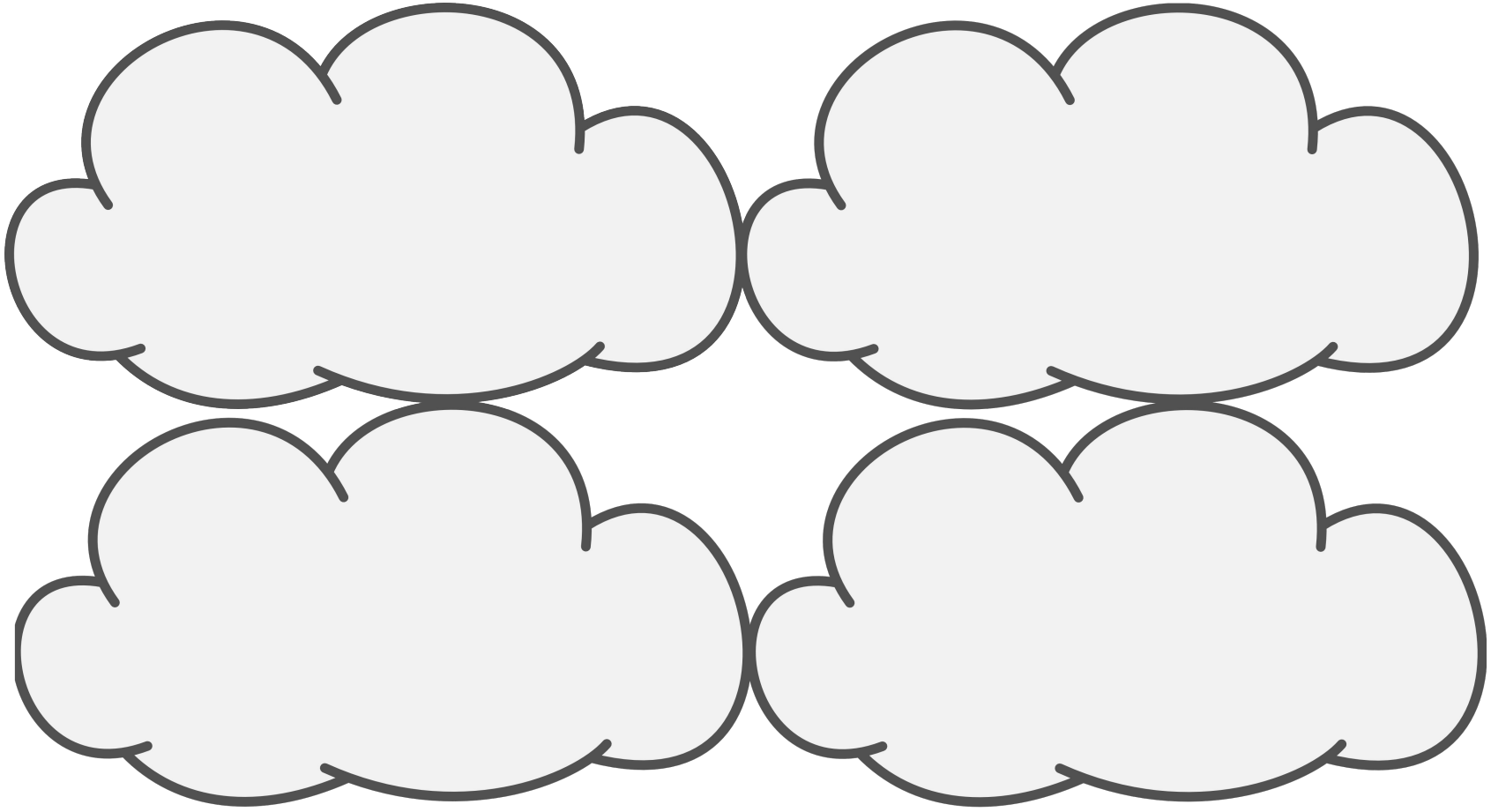
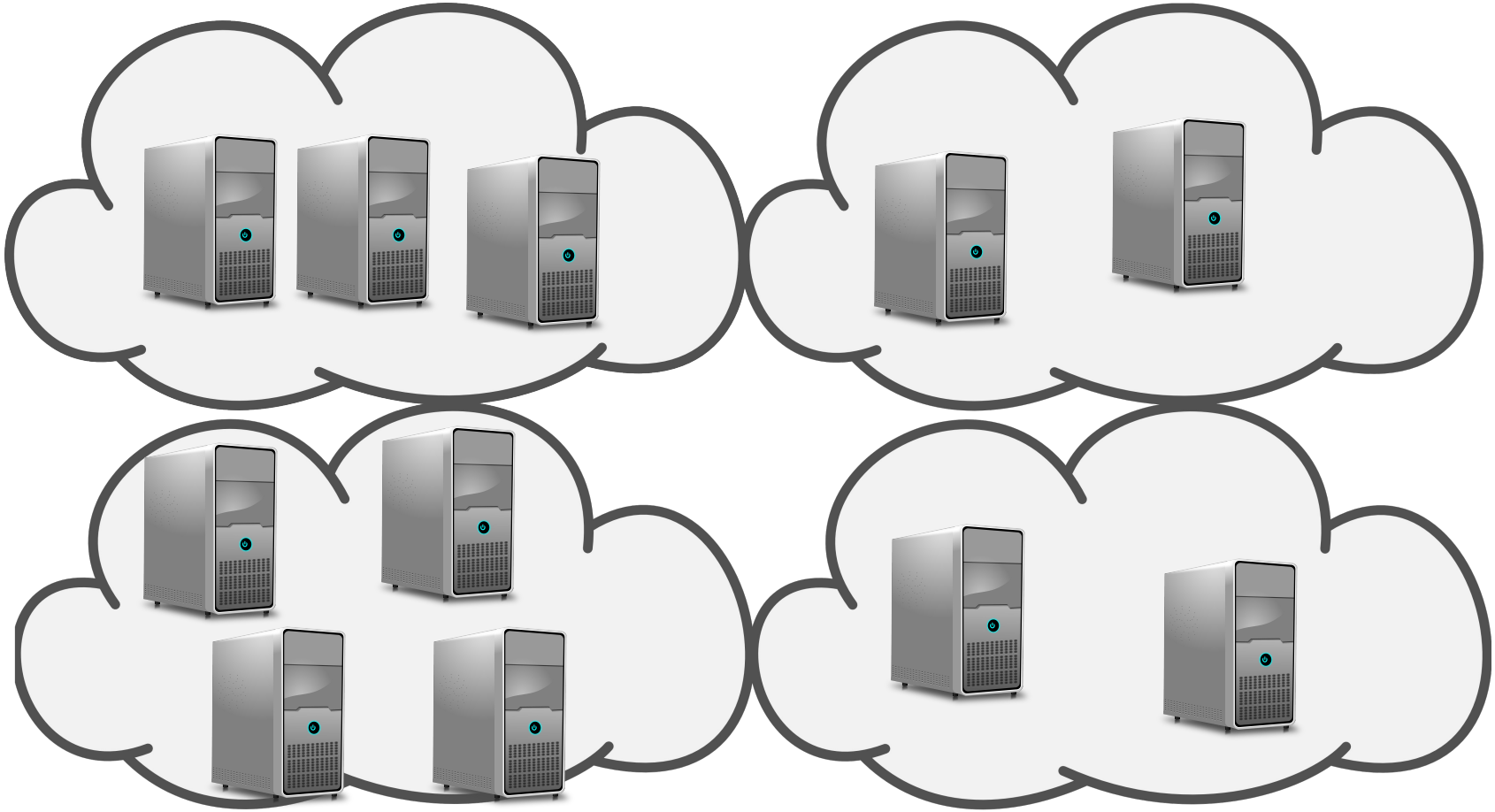
Cloud-aware programming

Nikolaos Bezirgiannis
CWI, Formal Methods
October 3, 2014

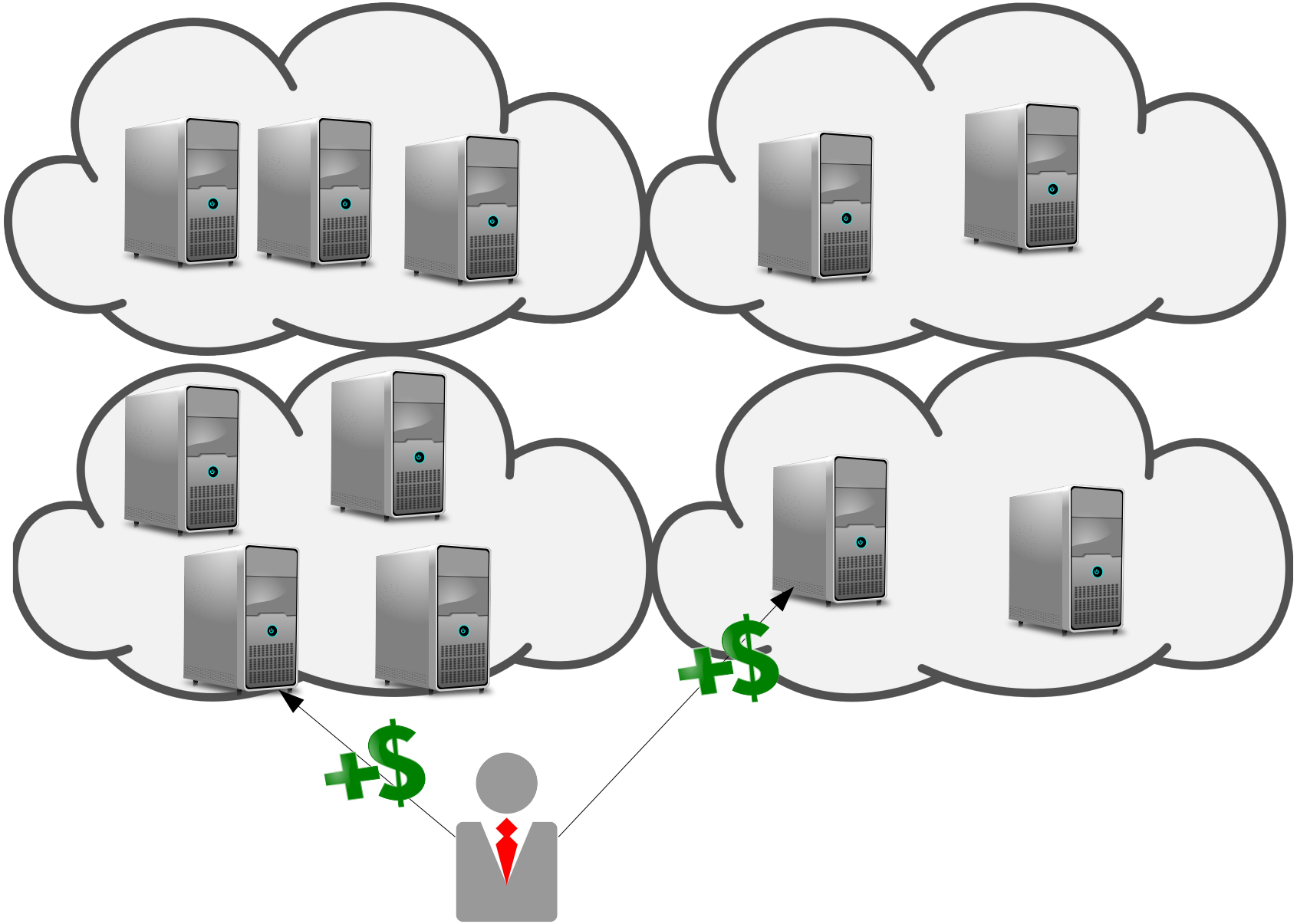
What is the Cloud?



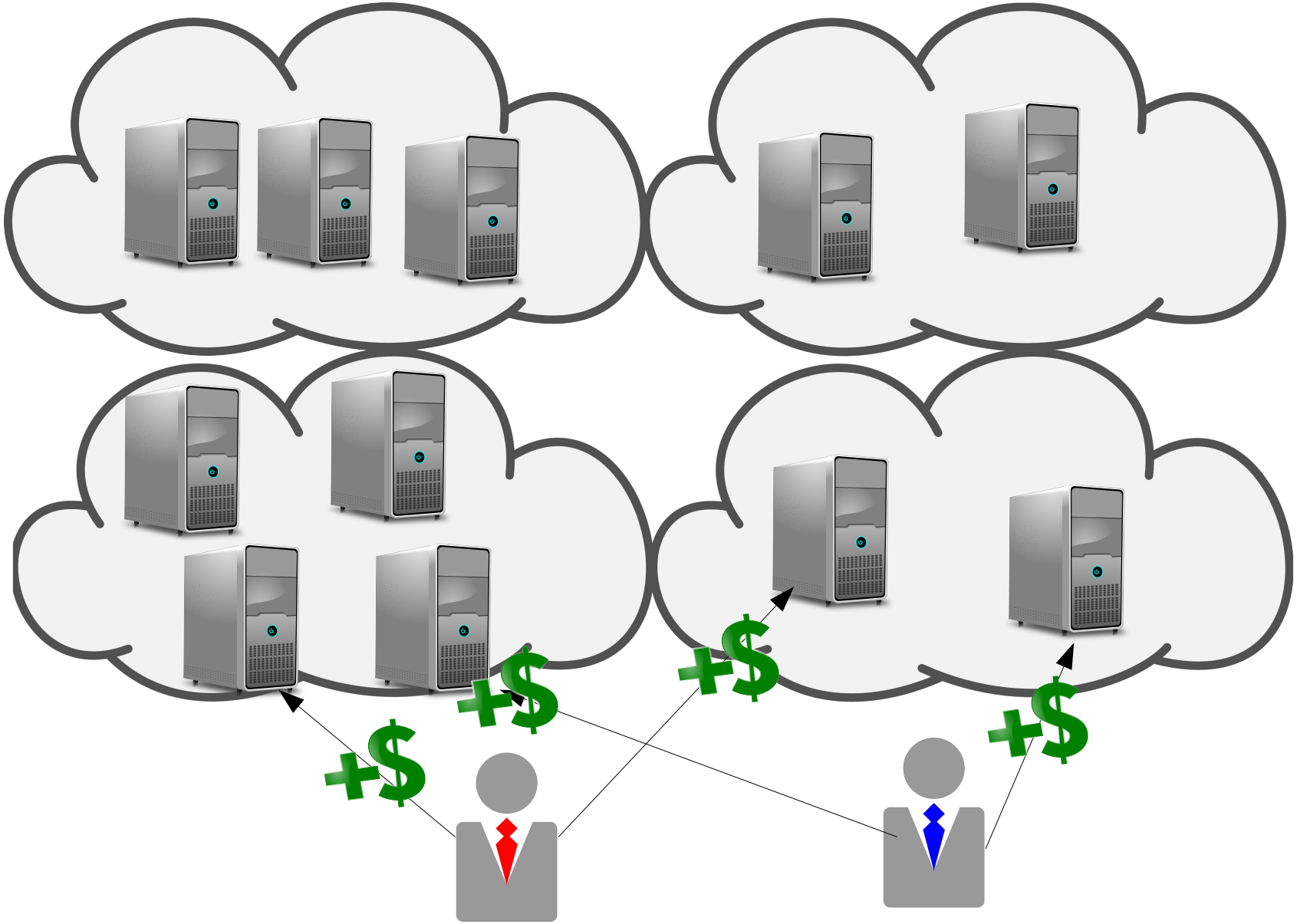
What is the Cloud?



What is the Cloud?



What is the Cloud?

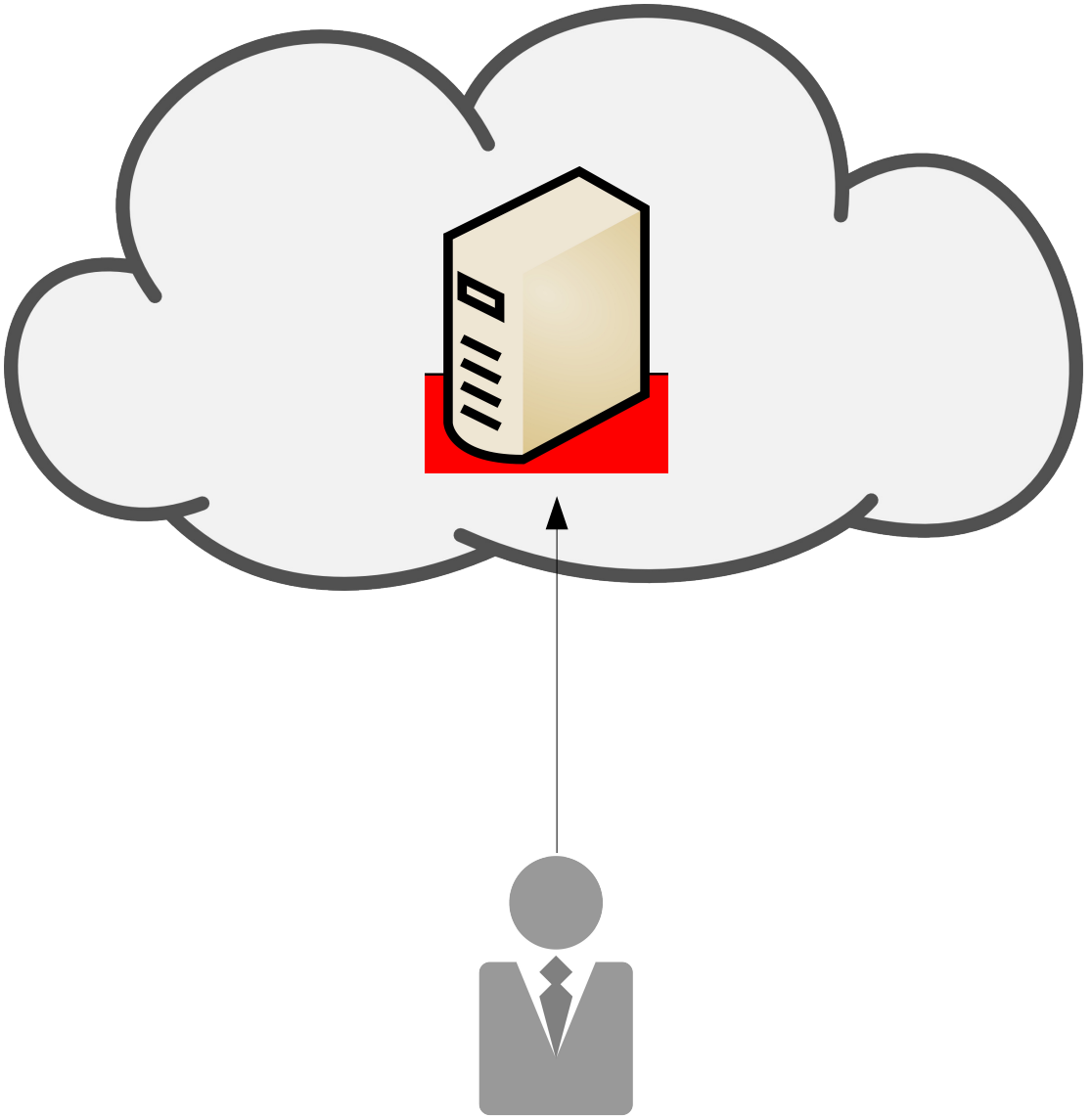


Why use the Cloud?

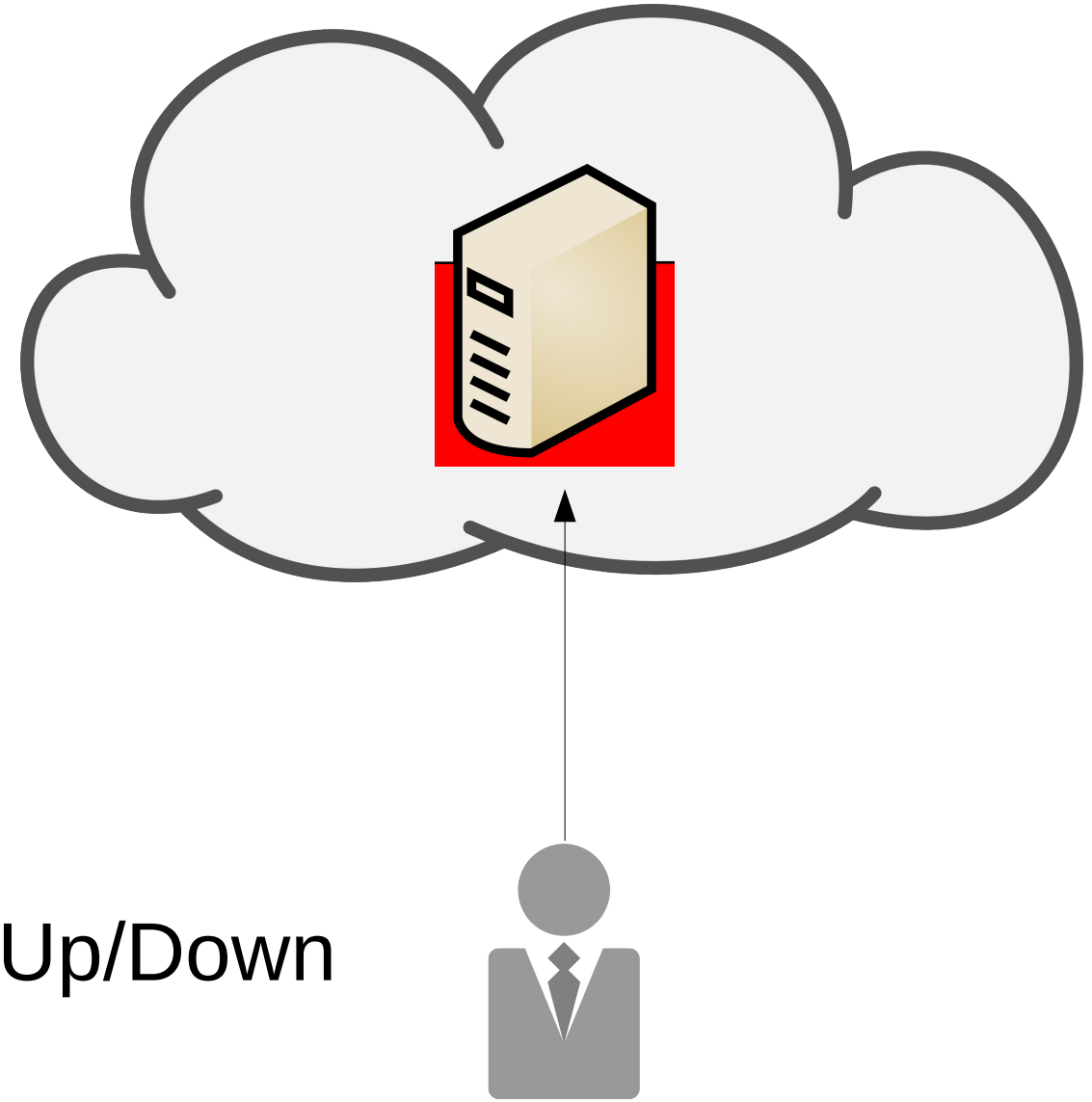
- Main reason:
  - More attractive economic model

- Elasticity
  - The ability of the Cloud to **scale** its resources based on the current **demand**.
    - Free unused resources and pay less.

Elasticity Example

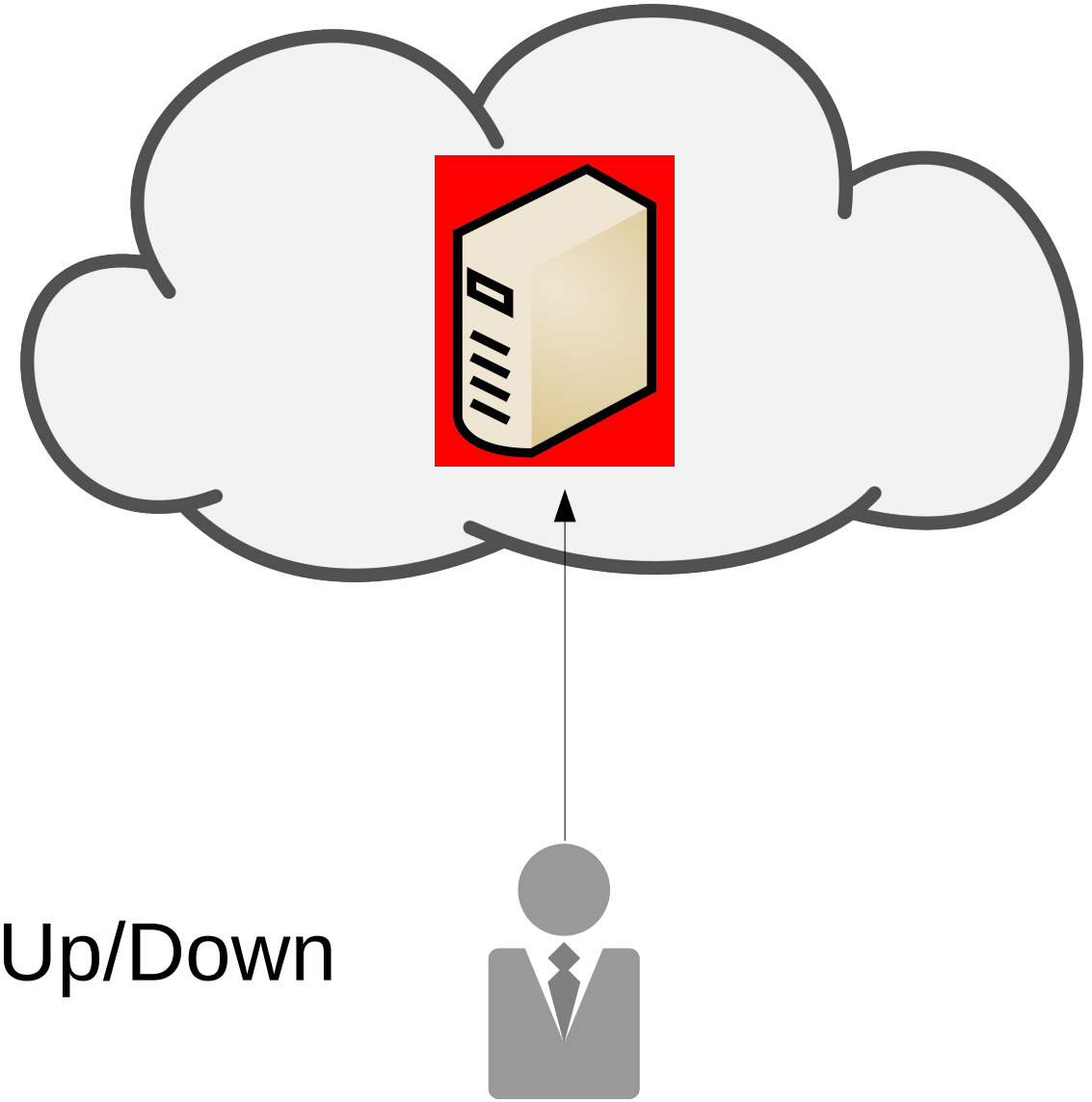


Elasticity Example



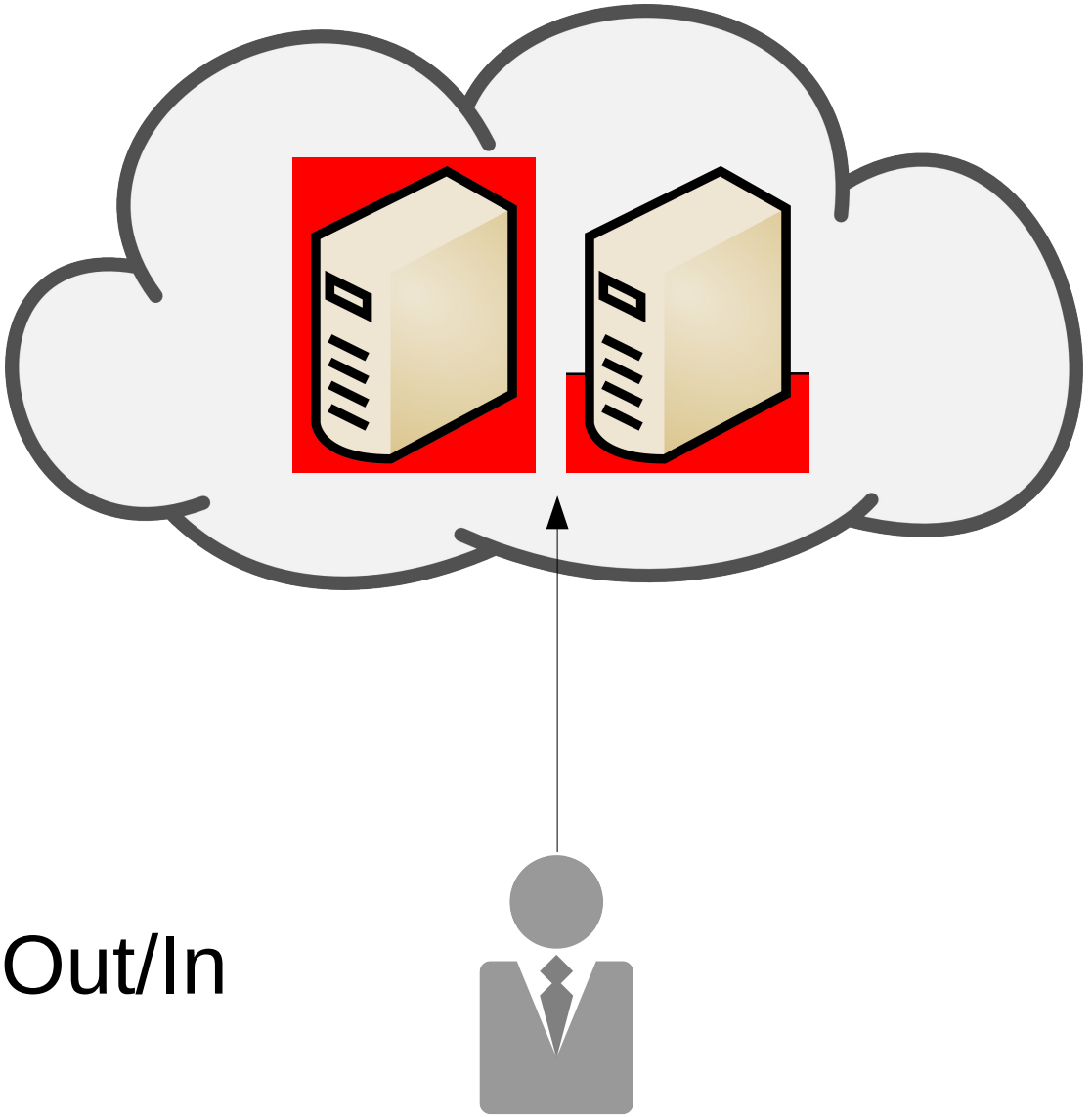
Scale Up/Down

Elasticity Example



Scale Up/Down

Elasticity Example

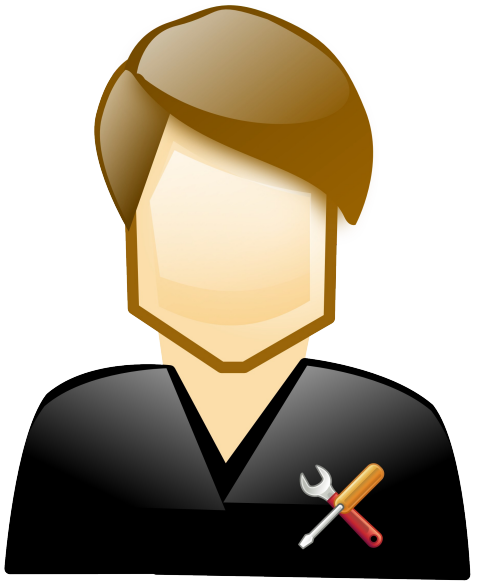


Scale Out/In

This talk

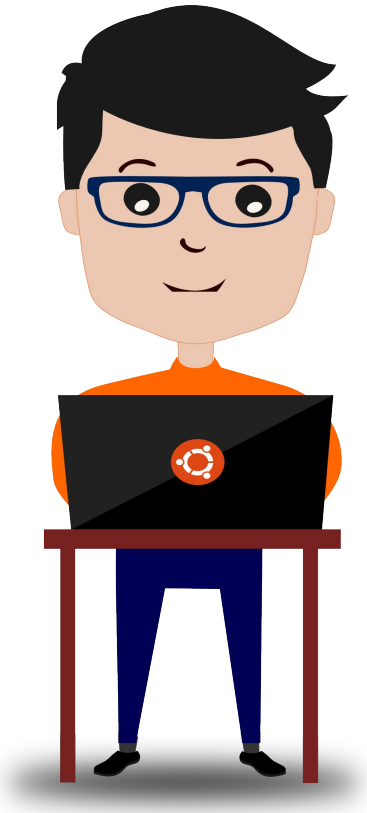
5 Technologies
for Elasticity

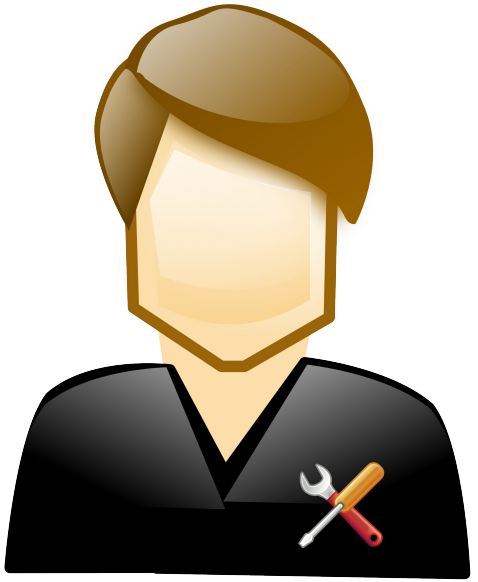
Technology #1



IT Operator/Administrator

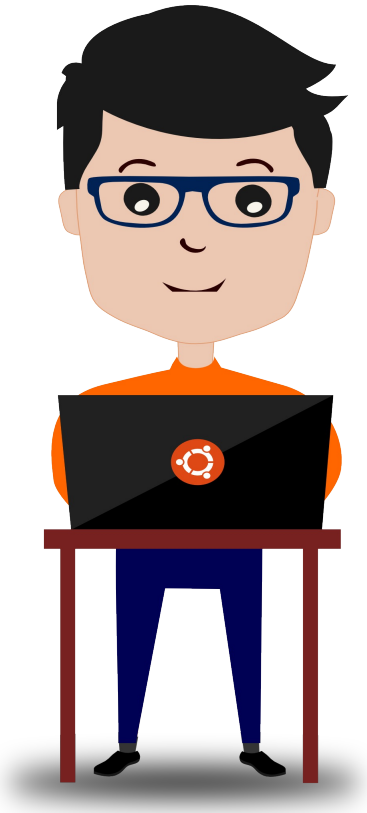
Technology #1

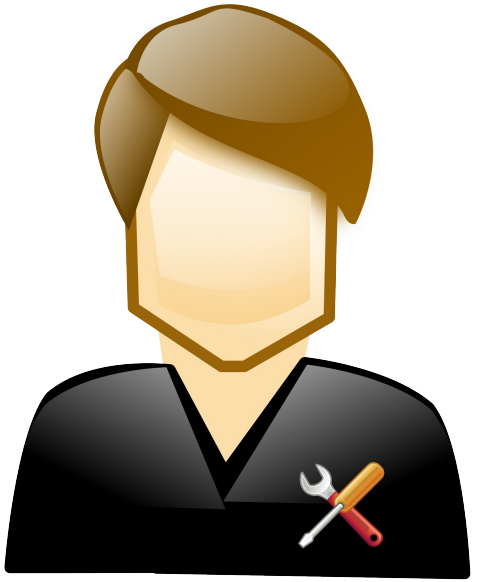




IT Operator/Administrator

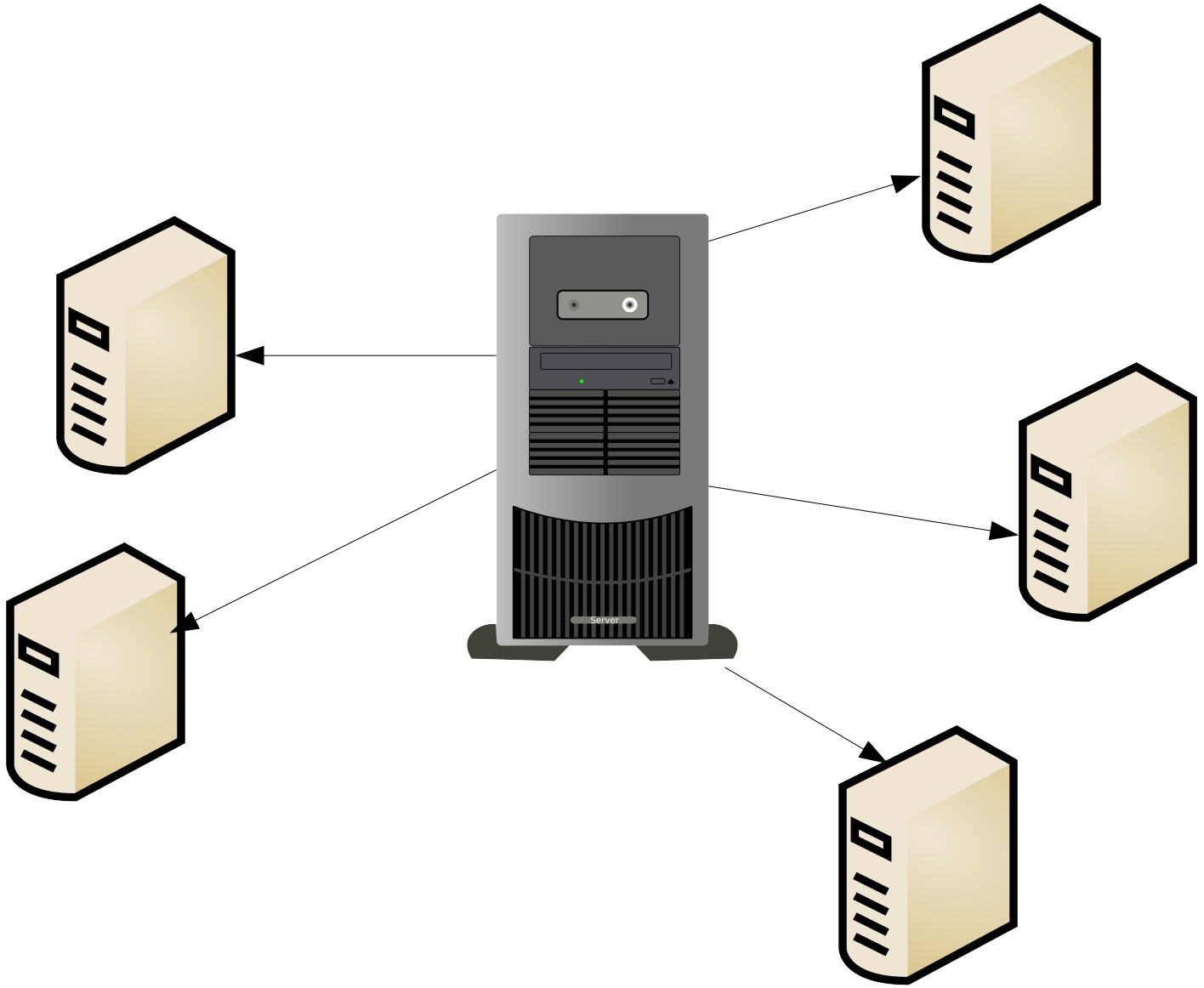
Technology #1





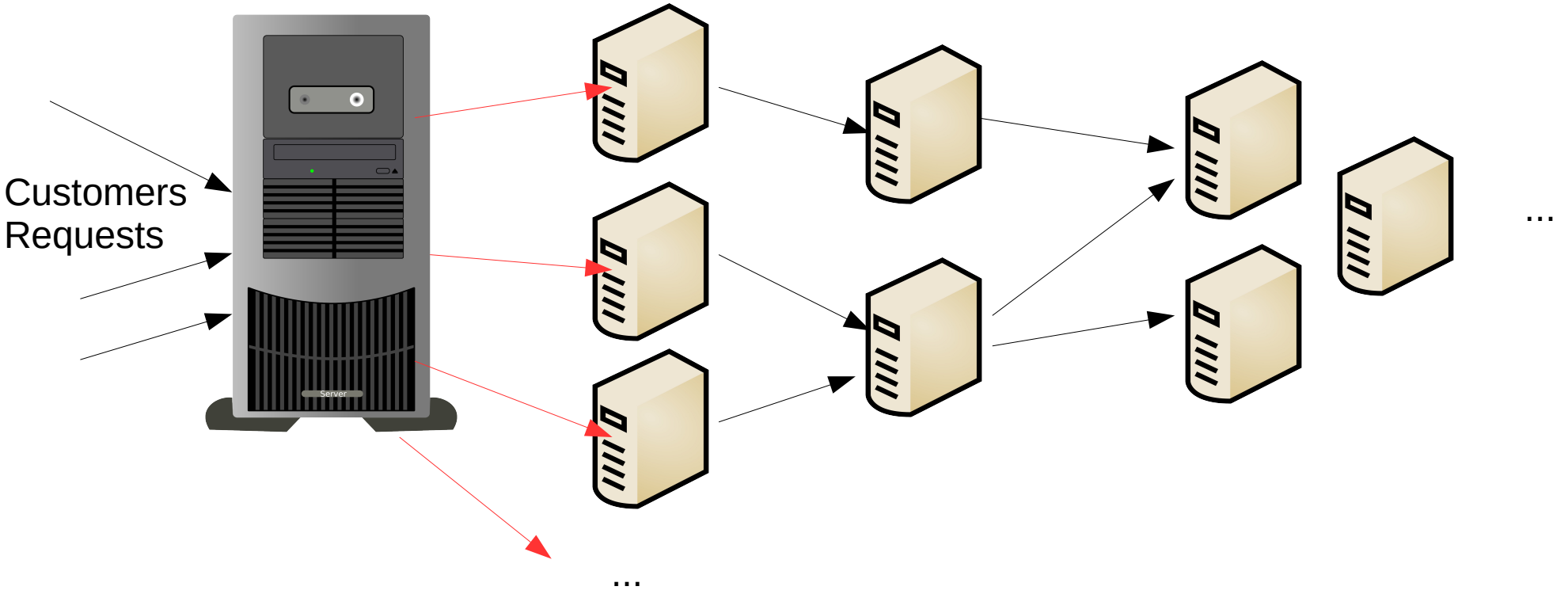
Does not scale well
Also humans are prone to errors!

Software to manage software



Technology #2
Load balancing

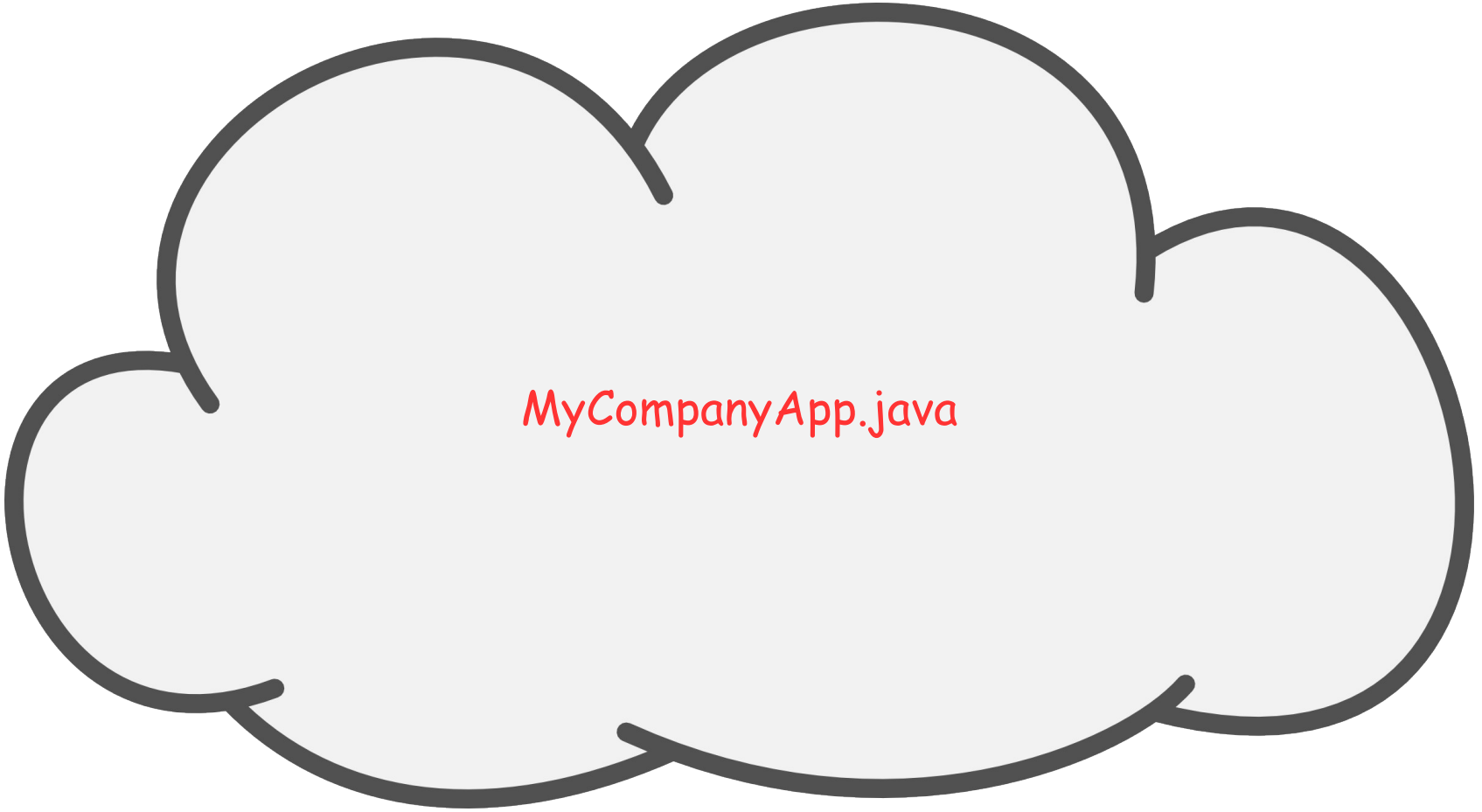
- Ancient technology but well-established



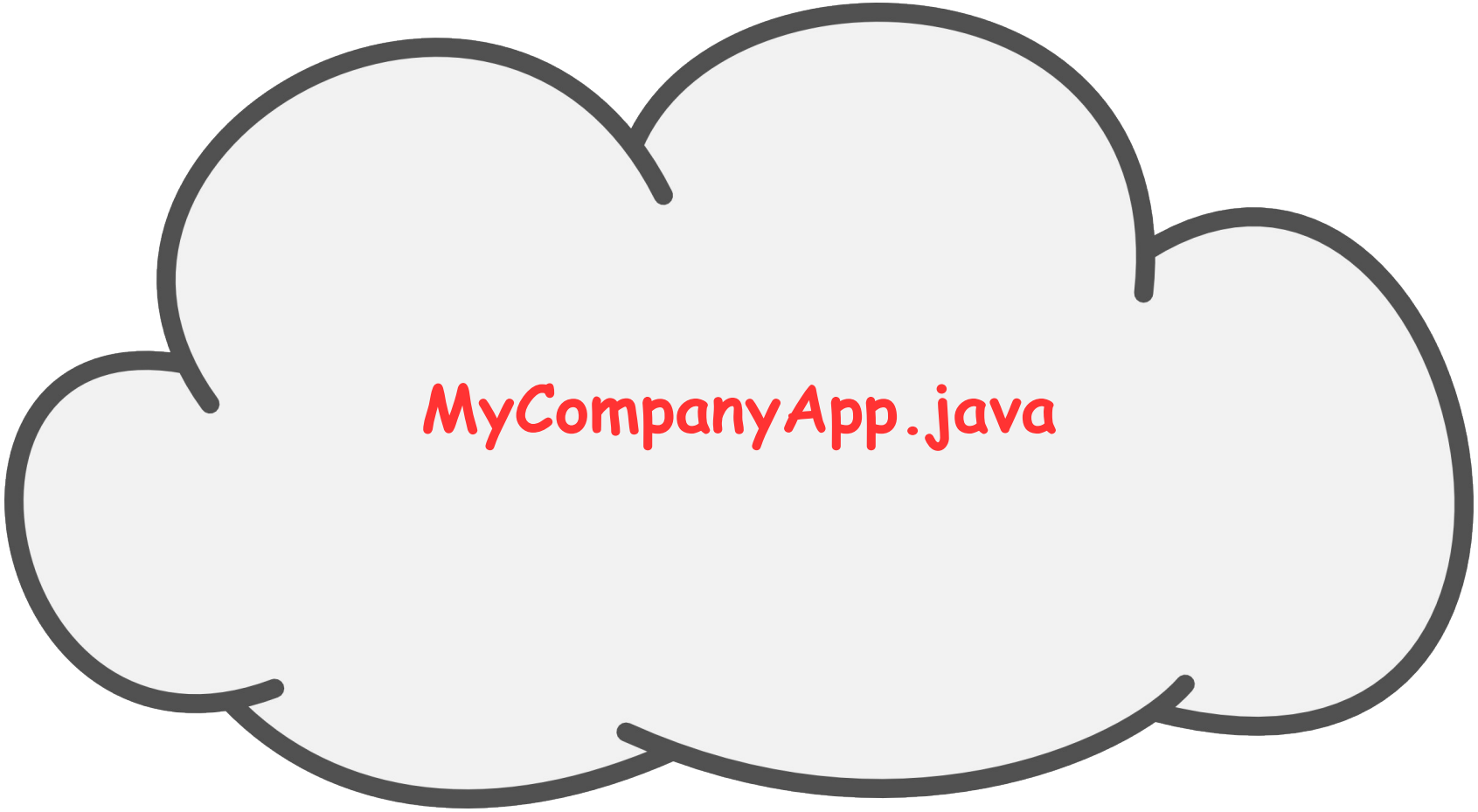
Technology #3
Cloud orchestration

- A central controller that (optimally) arranges the applications onto the cloud resources
- Upside
  - static analysis & dynamic monitoring
- Downside
  - Single point of failure
  - Difficult to describe custom **intelligence of** elasticity
- Example software:
  - Ubuntu Juju
  - OpenStack Heat

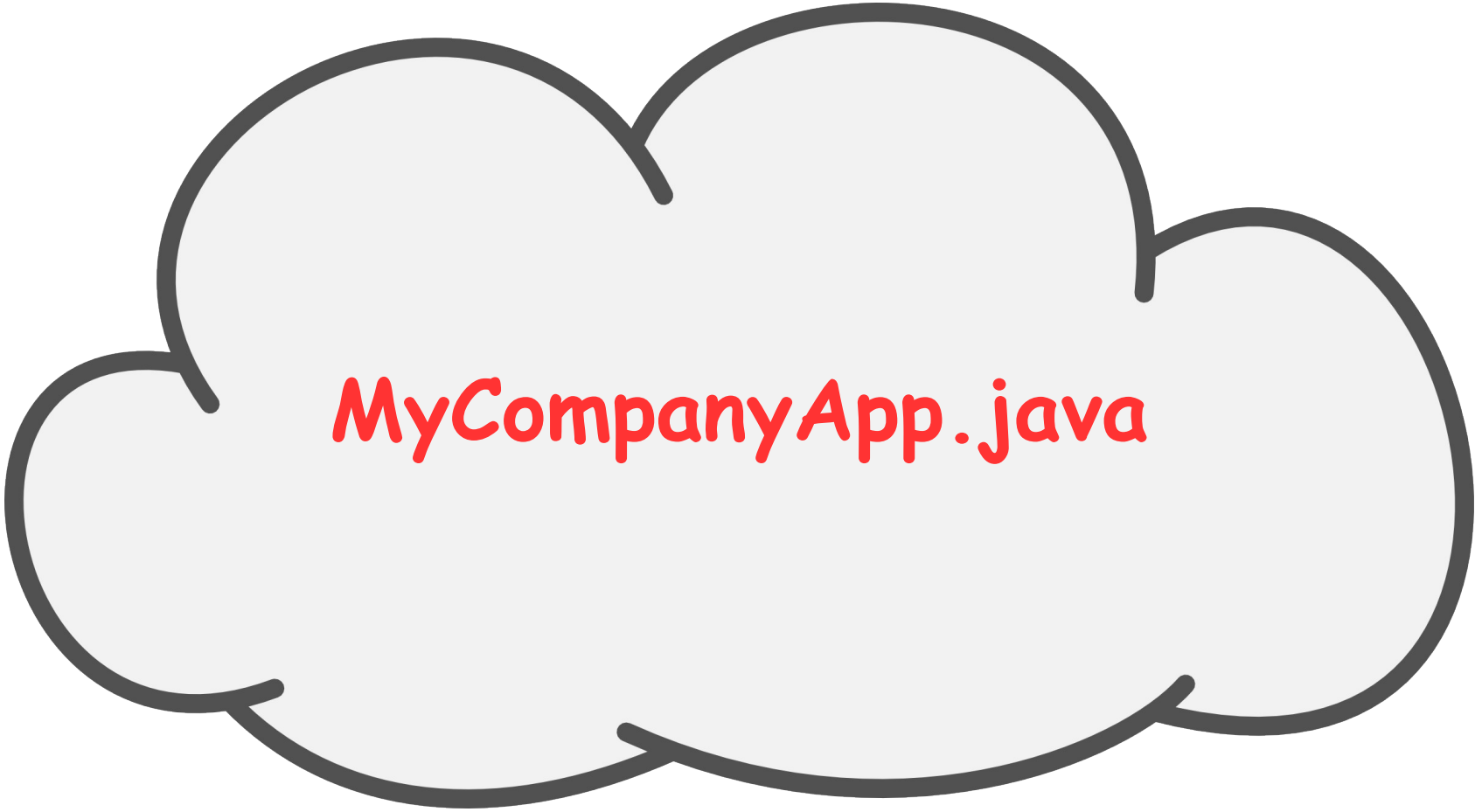
Technology #4
Platform-as-a-Service (PaaS)



Technology #4
Platform-as-a-Service (PaaS)



Technology #4
Platform-as-a-Service (PaaS)



Technology #4
Platform-as-a-Service (PaaS)

- Promising technology
- Example software/providers
  - Jelastic
  - CloudFoundry
  - Amazon ElasticBeansTalk
- Downside:
  - The technology does not scale out easily
  - Can be hard to migrate to different cloud provider

Our technology #5

- ABS programming language
  - Executable modeling language
  - Functional with Object-oriented charachteristics
  - Cooperative scheduling with **asynchronous methods**
  - **Distributed programming**, targeted specifically for the cloud
- Result of the **Envisage** European Project

Example snippet of ABS

```
DC dc1 = new DC (CPU(3) , MEM(8192))  ;
dcs = Cons(dc1, dcs);

Fut<List<Load>> avgs = map_load(dcs);

dc1 ! shutdown ();
```

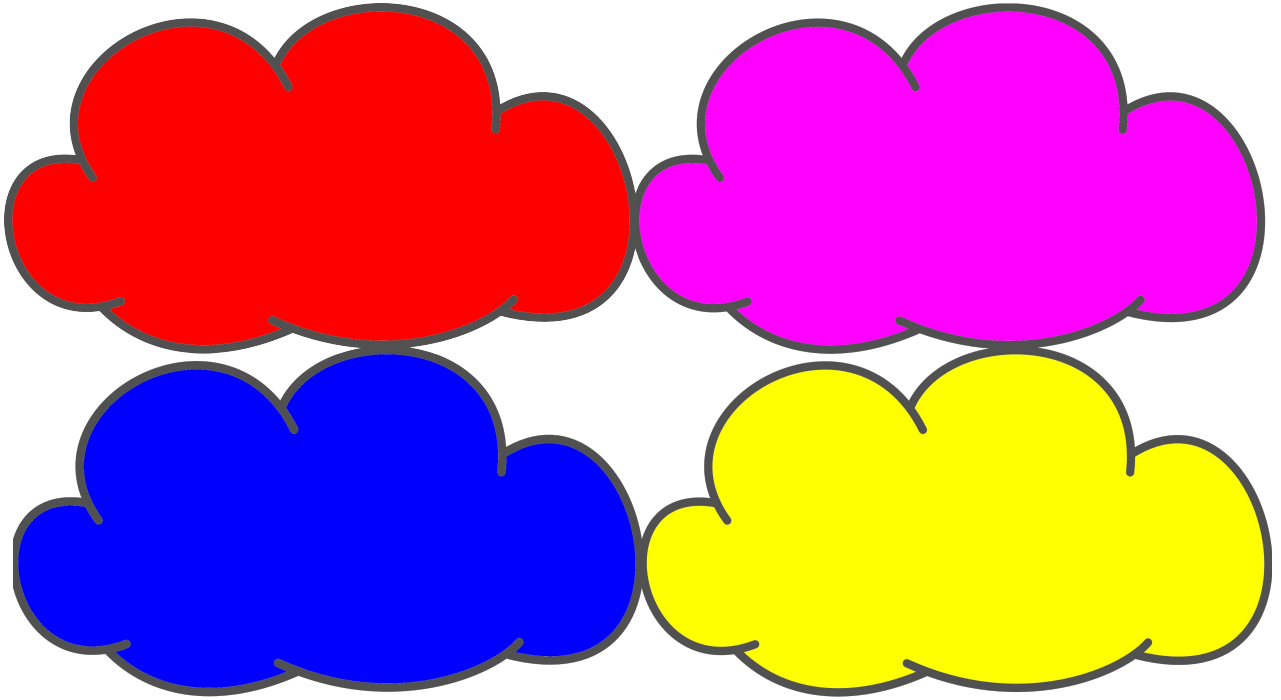
Example (continued)

Interf1 o1 = dc1 **spawns** Cls1 (params ..);

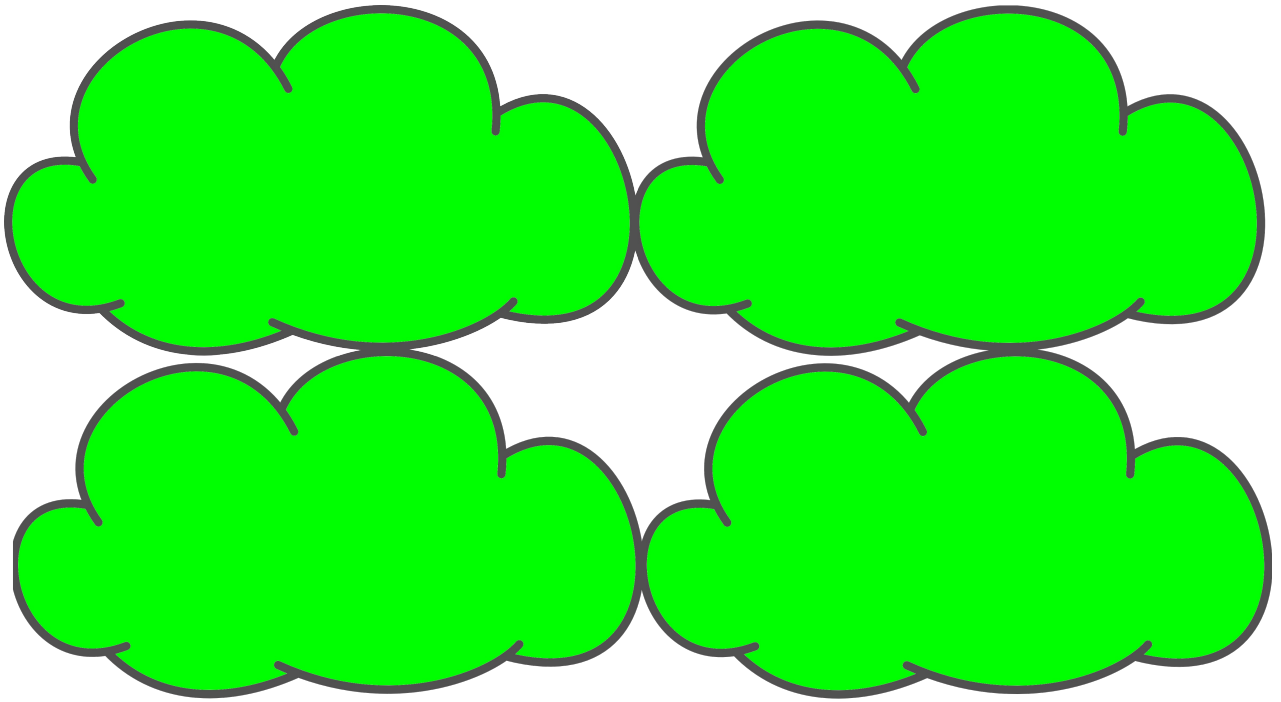
o1 ! method1 (params ..); // asynchronous

this . method2 (o1); // synchronous

Obstacle: Different cloud vendors



Solution: Unify Cloud interfaces



Existing solutions:
Apache jclouds, Redhat deltacloud
We use our own custom tool written in Haskell

Our Technology #5

- Upside
  - No single point of failure
  - Programmatically engineer **the logic** of **elasticity** and **provisioning**
  - Can include more elasticity metrics than **system load**
- Downside
  - Have to use our own ABS language
  - Unattractive to non-programmers?

Future Work

- Simulation of varying cloud deployments
- Incorporte Service-Level Agreements (SLA)
  - Static analysis
  - Monitoring

Links

- http://envisage-project.eu
- https://github.com/bezirg/abs2haskell