SELA|DEVELOPER|PRACTICE
Dec 2015

Bnaya Eshet & Avi Avni

Service Fabric
The next **Cloud Framework**

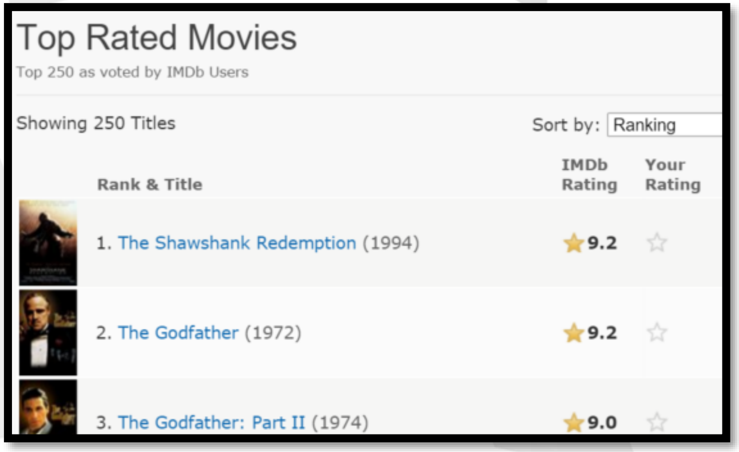
http://blogs.microsoft.co.il/blogs/bnaya/
http://blogs.microsoft.co.il/avi_avni
GitHub: https://github.com/bnayae/SDP-2015

SDK: https://azure.microsoft.com/en-us/documentation/articles/service-fabric-get-started/
User Voice: https://azure.microsoft.com/en-us/campaigns/service-fabric/

We need your help

- Google **IMDB** top 250
  use right click to copy URL

- **Twit** players and movie's URL
  with **#sdpf**



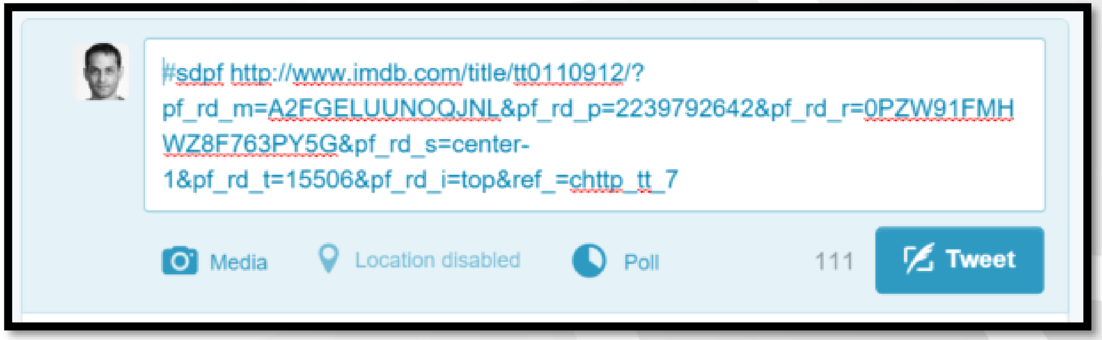
Top Rated Movies

Top 250 as voted by IMDb Users

Showing 250 Titles

Sort by: Ranking

Rank & Title	IMDb Rating	Your Rating
1. The Shawshank Redemption (1994)	★9.2	☆
2. The Godfather (1972)	★9.2	☆
3. The Godfather: Part II (1974)	★9.0	☆



#sdpf http://www.imdb.com/title/tt0110912/?pf_rd_m=A2FGELUUNOQJNL&pf_rd_p=2239792642&pf_rd_r=0PZW91FMHWZ8F763PY5G&pf_rd_s=center-1&pf_rd_t=15506&pf_rd_i=top&ref_=chttp_tt_7

Service Fabric

- The next **Cloud** (and On-Premise) **Framework**

In used by **Microsoft** over **several years**

- **Cortana** (500m evals/sec)
  thousand machine over different data center
- **Azure SQL Database** (1.4m databases)
- **Event Hub** (2bn event/day)
- **Skype For Businesses, Azure Service Bus, Azure Document DB, Power BI, and more…**

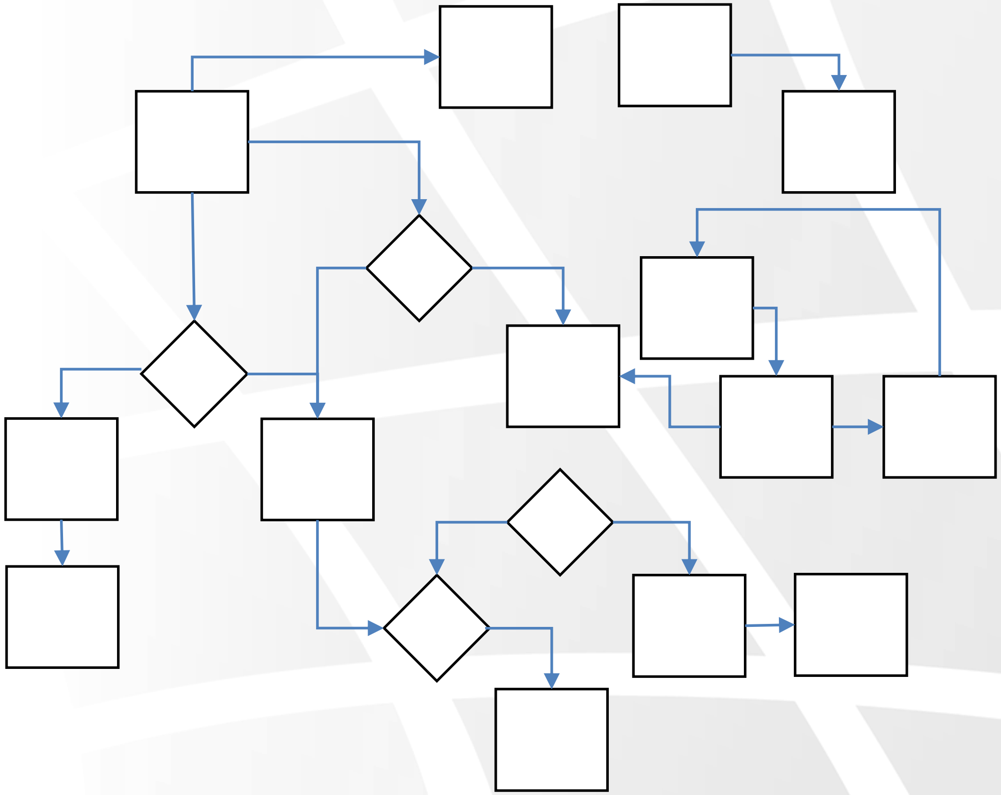
Microsoft® SQL Azure™

Agenda

- **Why** application gets so complex?
- **How** to make **complex** system **simple**?
- **APIs** and **Basic Concepts**
- **Demo**
- **Patterns**

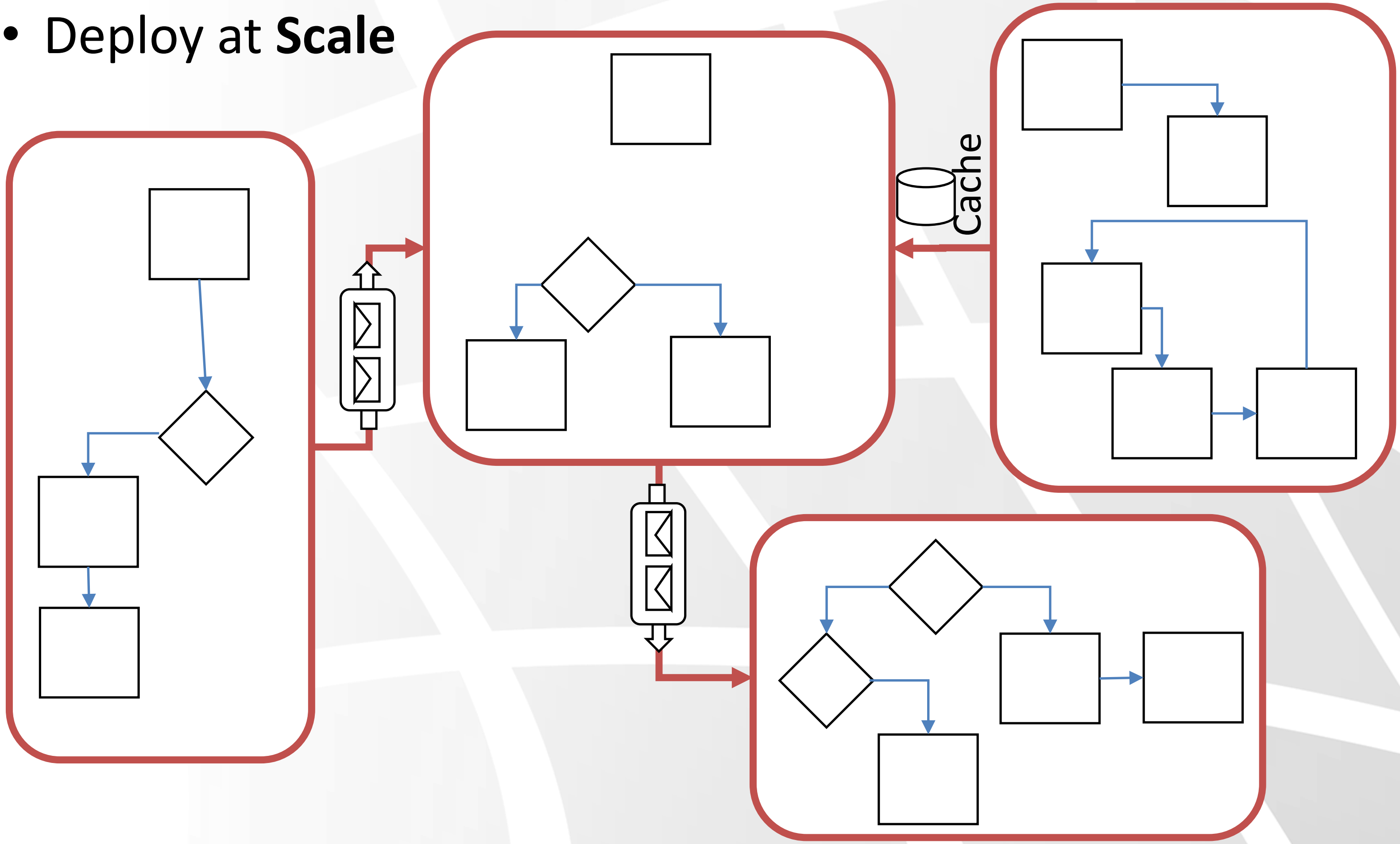
Service Fabric

- **Complex** problem can be describe as set of **Simple** Problem

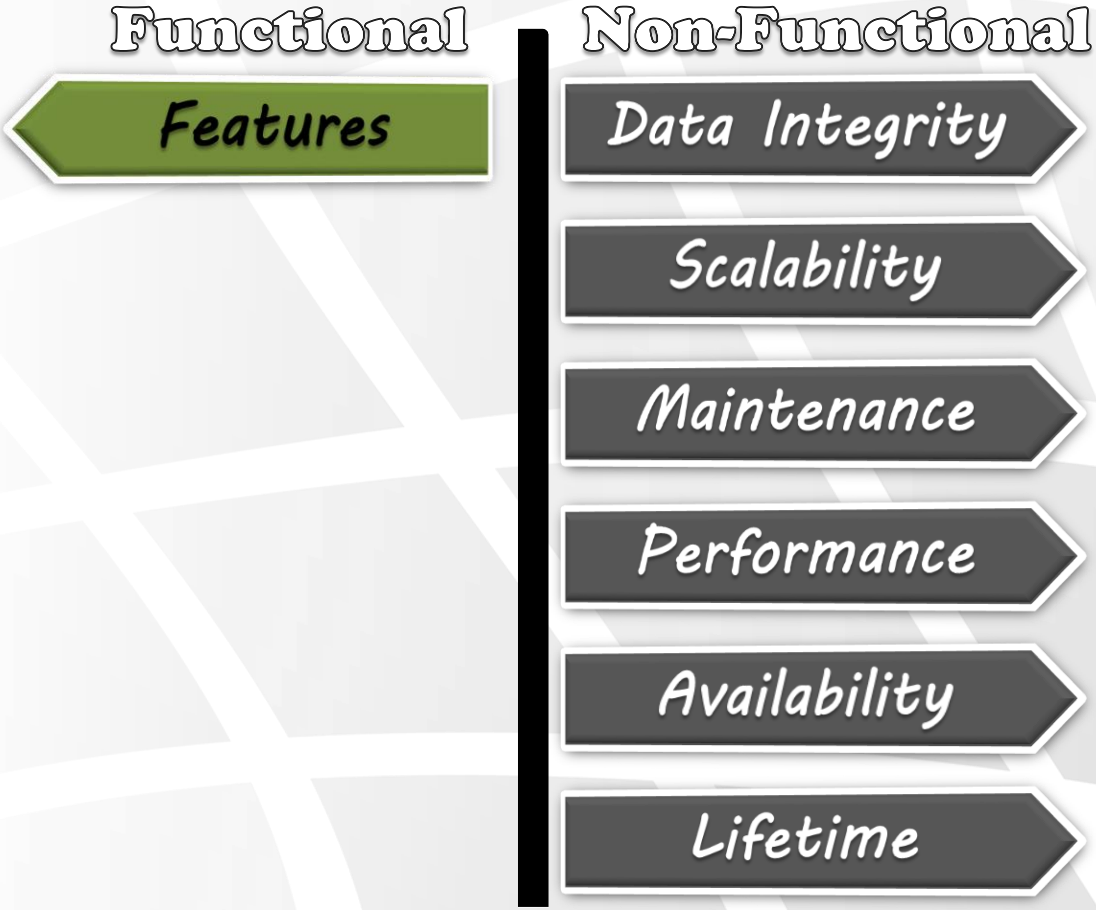


Service Fabric

- **Complex** problem can be describe as set of **Simple** Problem
- Deploy at **Scale**



Modern App Development



Functional

Features

Non-Functional

Data Integrity

Scalability

Maintenance

Performance

Availability

Lifetime

Paradigms

OO / Component Oriented

Multi Tier

SOA

Web
Role

Worker
Role

Worker
Role

Service Fabric

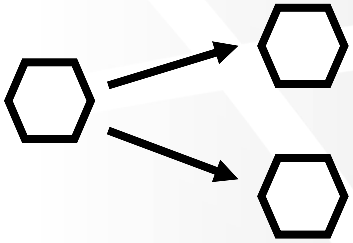
Coding Model

- **Reliable Services** (kind of Advance Web / Worker Roles)
- **Actor**
  - Encapsulate both **Behavior** and **State** ≈ **class**
  - Communicate via **Messaging** ≈ **invoke method**
  - **Lifetime** controlled by the **Fx** ≈ **.NET GC**
    Virtual (Lazy) **Allocation** ≈ **IoC**

Coding Model

- **Reliable Services** (kind of Advance Web / Worker Roles)
- **Actor**
  - Encapsulate both **Behavior** and **State** ≈ **class**
  - Communicate via **Messaging** ≈ **invoke method**
  - **Lifetime** controlled by the **Fx** ≈ **.NET GC**

    Virtual (Lazy) **Allocation** ≈ **IoC**



Inspiration

Orleans (Microsoft Research)
https://github.com/dotnet/orleans

AKKA.NET (Open Source)
https://github.com/akkadotnet/akka.net

Actor

Contract + Implementation

```
public interface IParserActor : IActor
{
    Task<Movie> ParseAsync(string html);
}
```

```
public class ParserActor : StatelessActor, IParserActor
{
    public Task<Movie> ParseAsync(string html) {...}
}
```

Actor

Virtual (Lazy) **endpoints** is acting like kind of **distributed IoC**

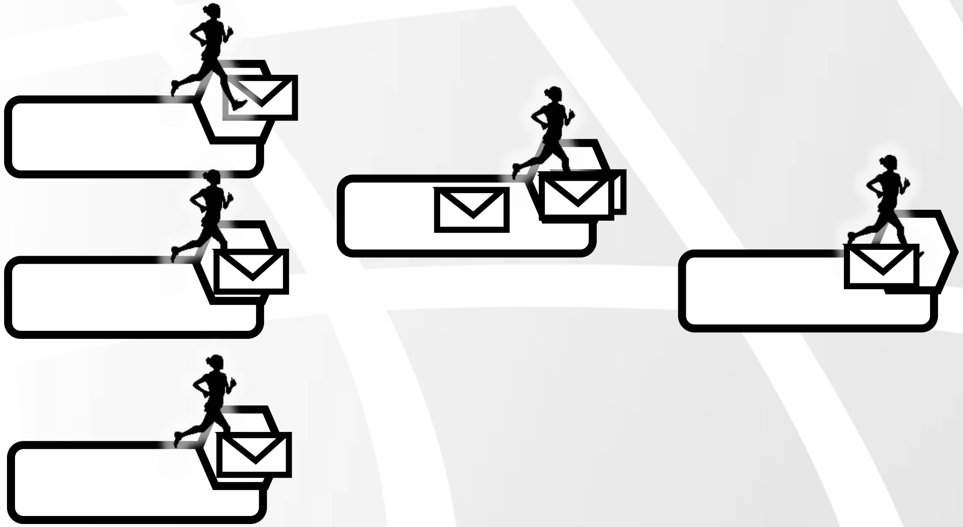
```
using (var fabricRuntime = FabricRuntime.Create())
{
    fabricRuntime.RegisterActor(typeof(ParserActor));
    //...
}
```

```
var actorId = new ActorId(Guid.NewGuid());
IParserActor parser =
     ActorProxy.Create<IParserActor>(actorId);

Movie result = await parser.ParseAsync("<HTML>...</HTML>");
```

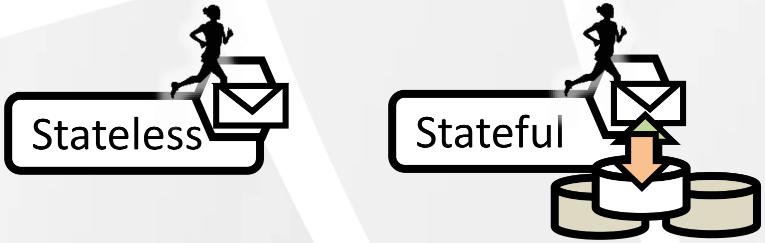
Turn-based Concurrency

Actor is **Single Thread** Component



Stateful Actor

- Actor encapsulate **Behavior** and **State**



Stateful Actor

- Actor encapsulate **Behavior** and **State**

```
[DataContract]
public class Token
{
    [DataMember]
    public string Data { get; set; }
}
```

Stateful Actor

```
public class UserActor : StatefulActor<Token>, IUserActor
{
    public override async Task OnActivateAsync()
    {
        if(State.Data != null)
       {
               string token = await Facebook.LoginAsync(…);
               State.Data = token;
        }
    }

    public Task<string> GetToken()
    {
        return Task.FromResult(State.Data);
    }
}
```

Actor’s Events

- Actor to Service pub-sub

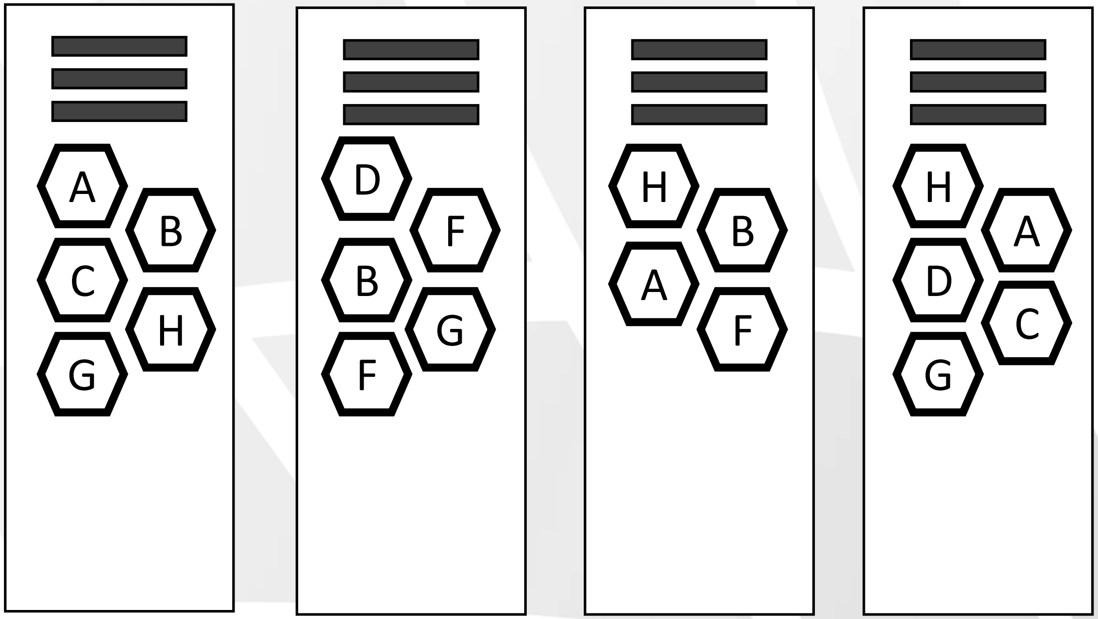
```
public interface IImdbEvents : IActorEvents
{
    void Changed(TwittData  data);
}
```

```
public interface IImdbHub : IActor
    ,IActorEventPublisher<IImdbEvents>
{
    …
}
```

```
var proxyHub = ActorProxy.Create<IImdbHub>(hubId);
await proxyHub.SubscribeAsync<IImdbEvents>(this);
```

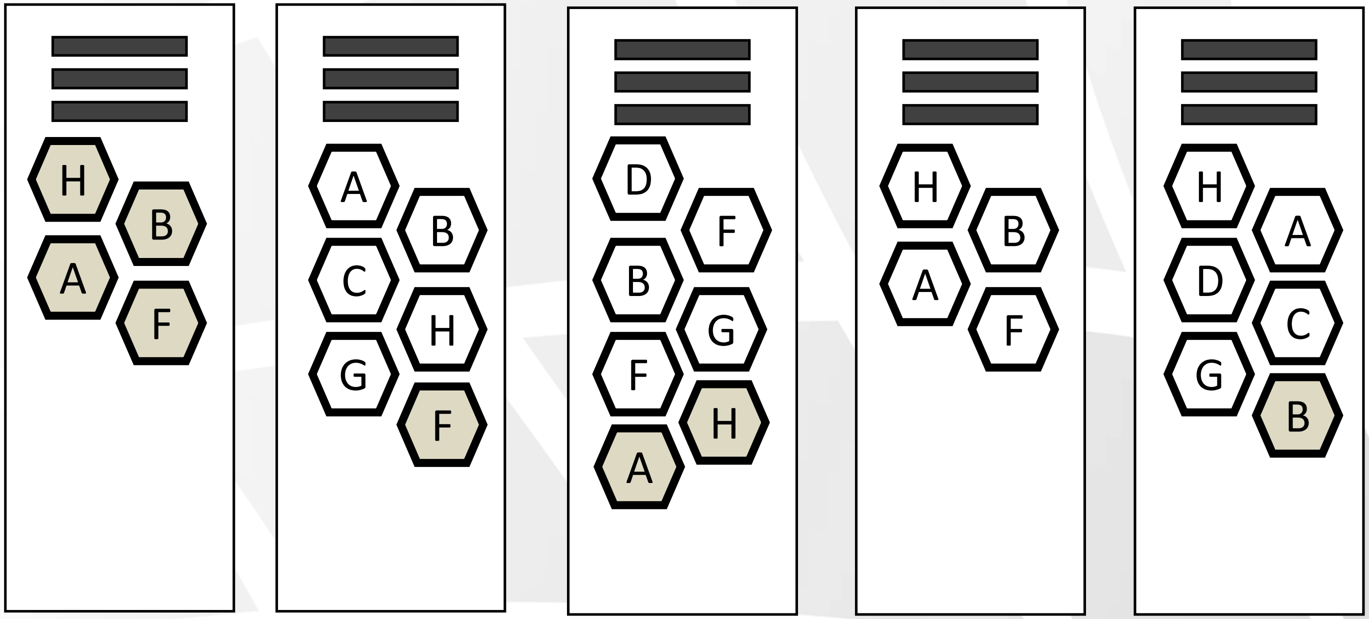
Scalability

- Automatic Deployment



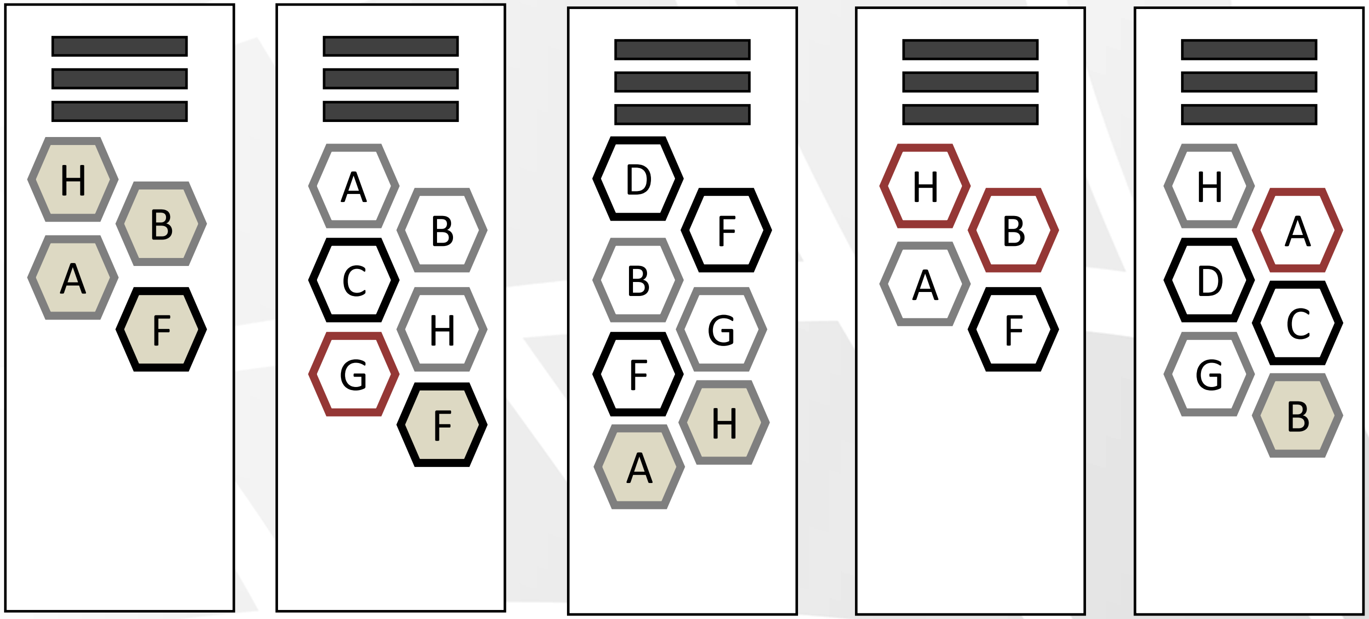
Deployment

- High Availability
- Fault Tolerance
- Scalability



Deployment HA and Fault Tolerance

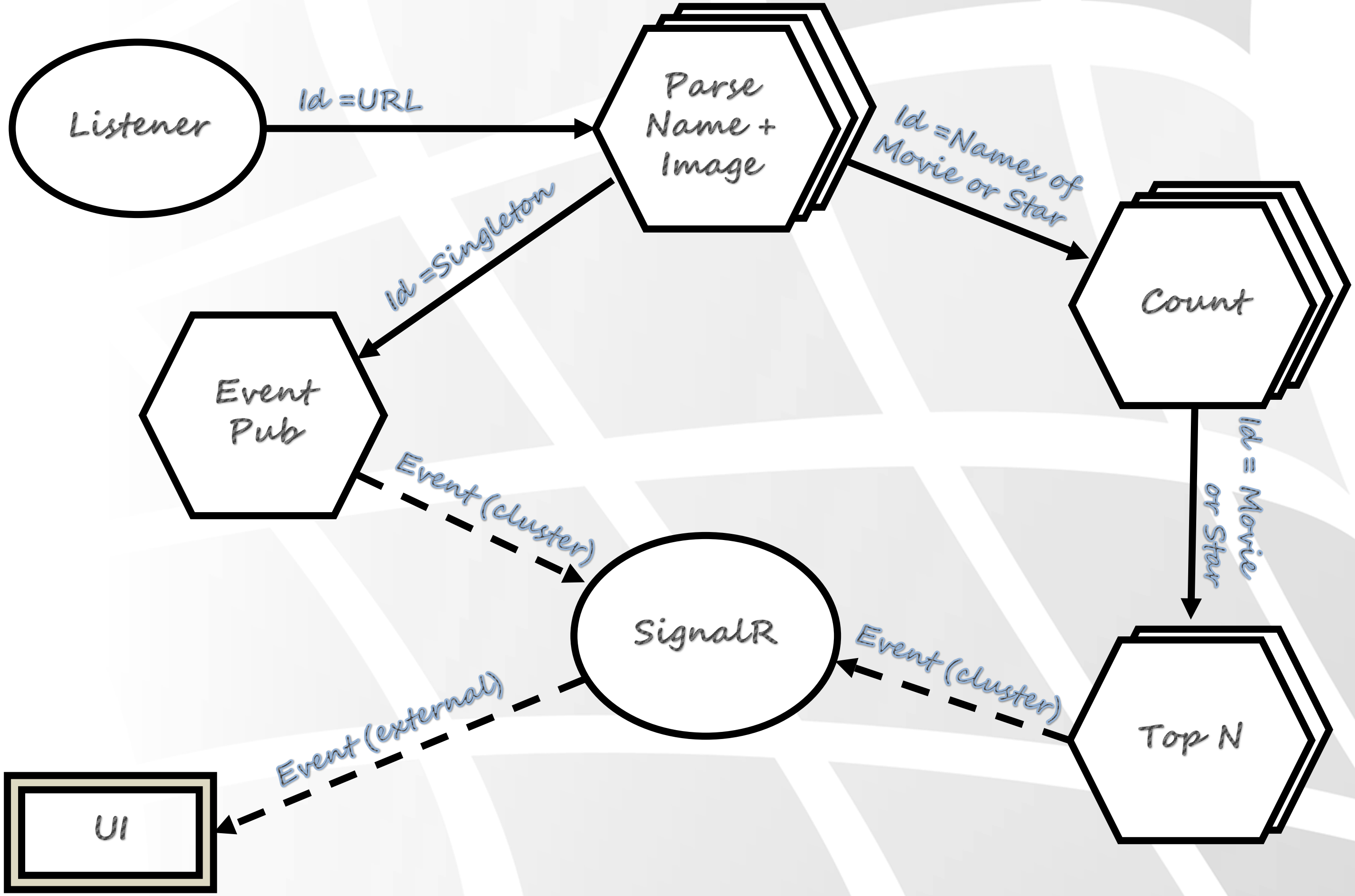
Stateful Model



Summary

- **Single Responsibility** (Easy Maintenance)
- **Thread Safety** (Turn-based Concurrency)
- Built-in **Queue**
- **Stateful** and **Stateless** model
- Internal **Communication** (IoC like)
- **Dynamic Balanced Deployment** (and **Upgrade**)
- **Resilient to Failure** and **High Availability**

Demo

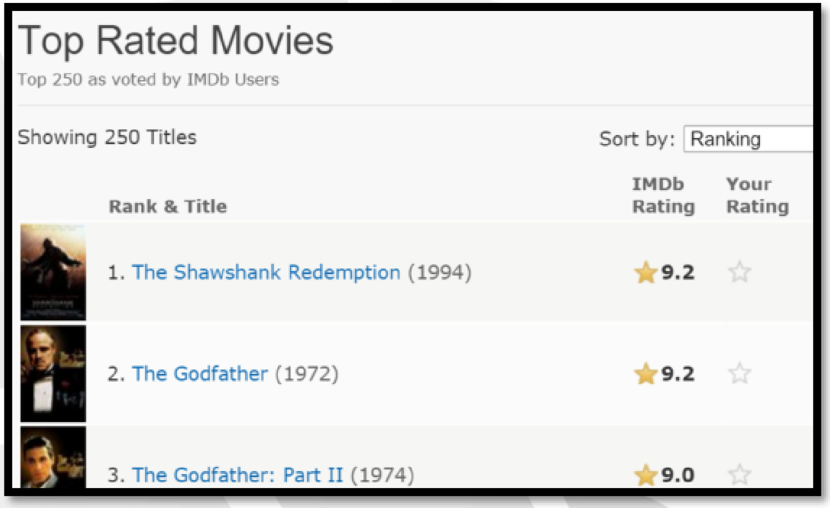

Listener
Id =URL
Parse Name + Image
Id =Names of Movie or Star
Id =Singleton
Count
Event Pub
Id = Movie or Star
Event (cluster)
SignalR
Event (cluster)
Top N
Event (external)
UI

Demo

We need your help

- Google **IMDB** top 250
  use right click to copy URL

- **Twit** players and movie’s URL
  with **#sdpf**



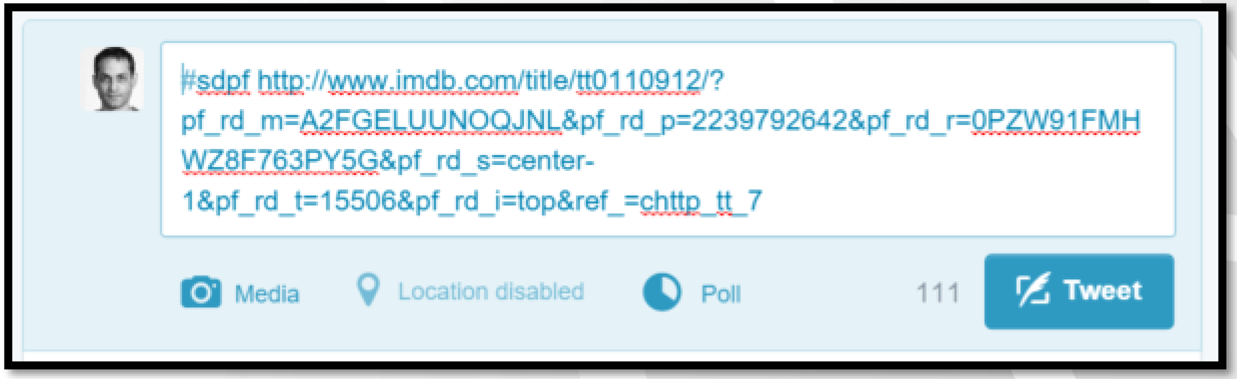
Top Rated Movies

Top 250 as voted by IMDb Users

Showing 250 Titles

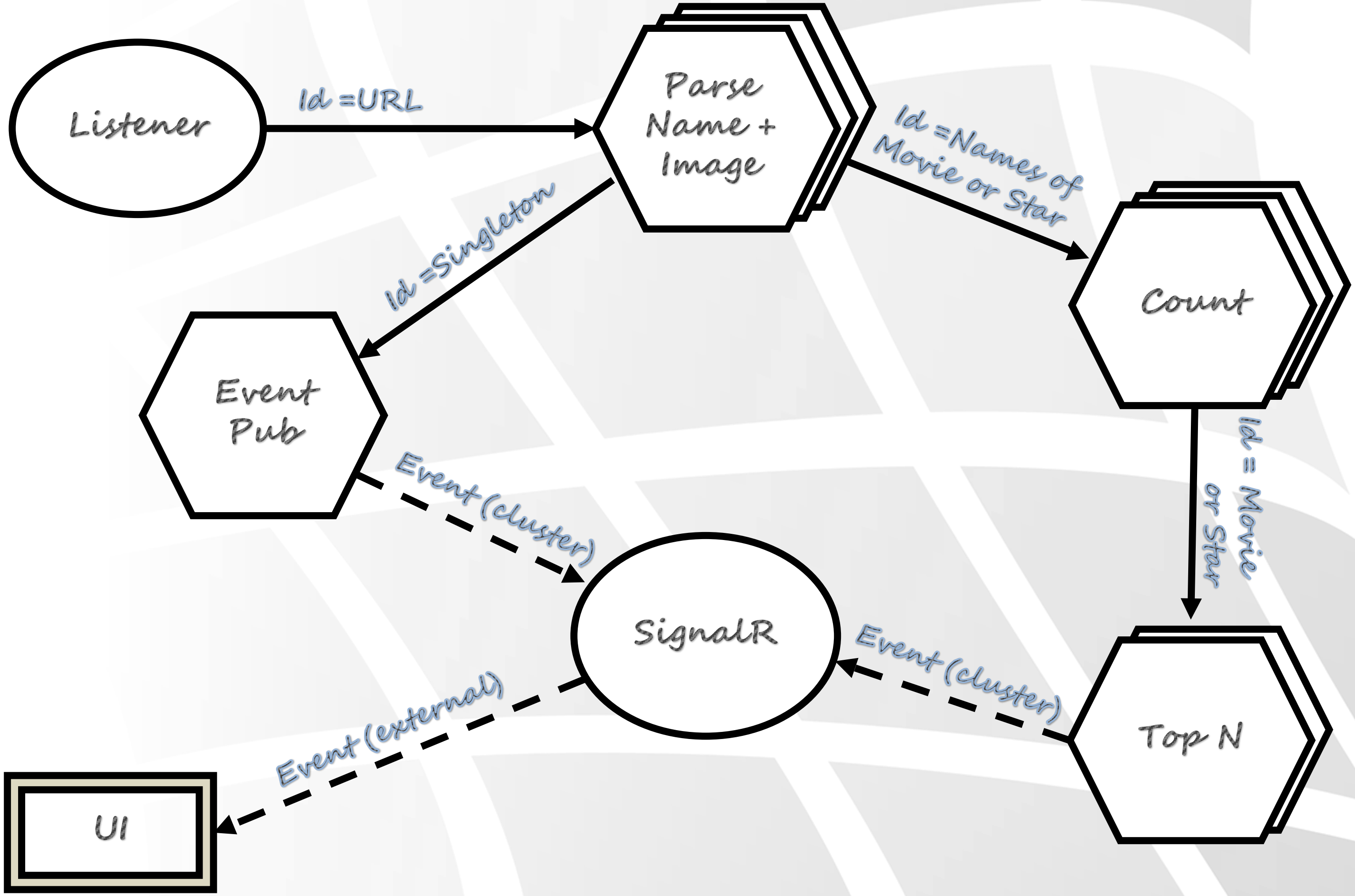
Sort by: Ranking

Rank & Title	IMDb Rating	Your Rating
1. The Shawshank Redemption (1994)	★9.2	☆
2. The Godfather (1972)	★9.2	☆
3. The Godfather: Part II (1974)	★9.0	☆



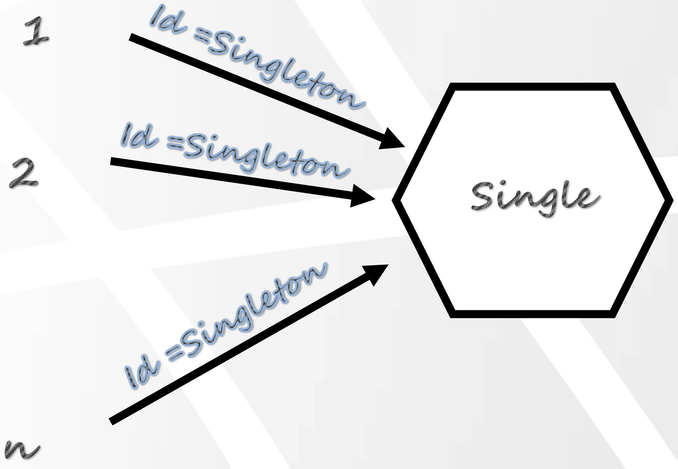
#sdpf http://www.imdb.com/title/tt0110912/?pf_rd_m=A2FGELUUNOQJNL&pf_rd_p=2239792642&pf_rd_r=0PZW91FMHWZ8F763PY5G&pf_rd_s=center-1&pf_rd_t=15506&pf_rd_i=top&ref_=chttp_tt_7

Demo


Listener
Id =URL
Parse Name + Image
Id =Names of Movie or Star
Count
Id =Singleton
Event Pub
Id = Movie or Star
Event (cluster)
SignalR
Event (cluster)
Top N
Event (external)
UI

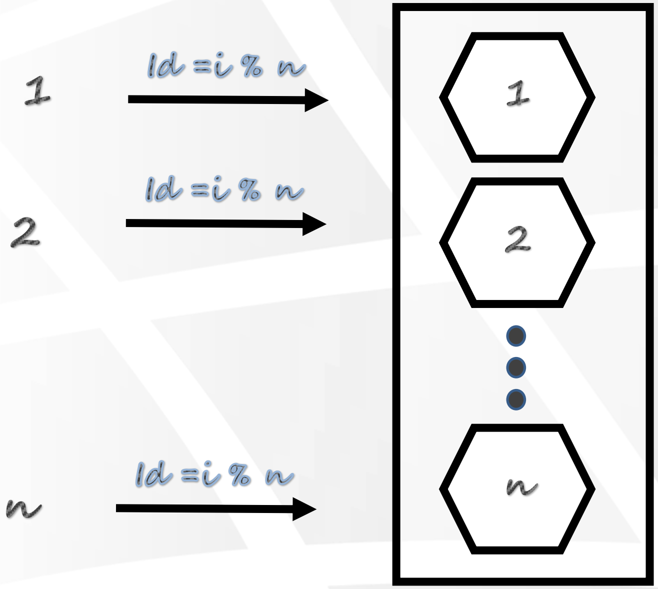
Id = Pattern

- Singleton pattern



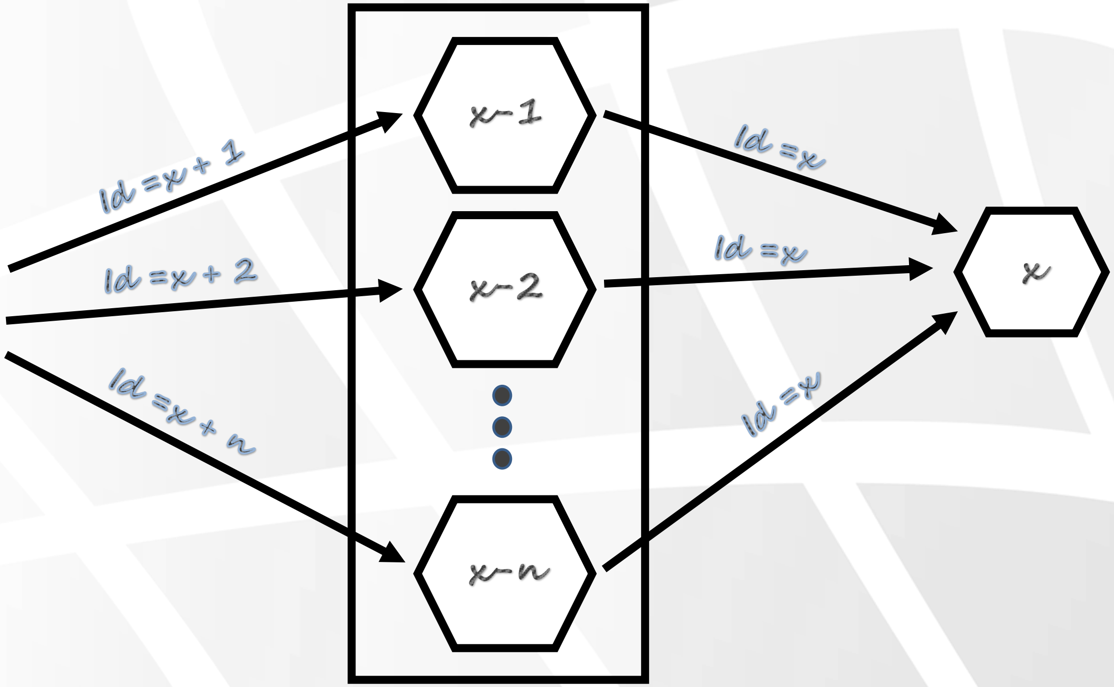
Id = Pattern

- Which pattern is this?



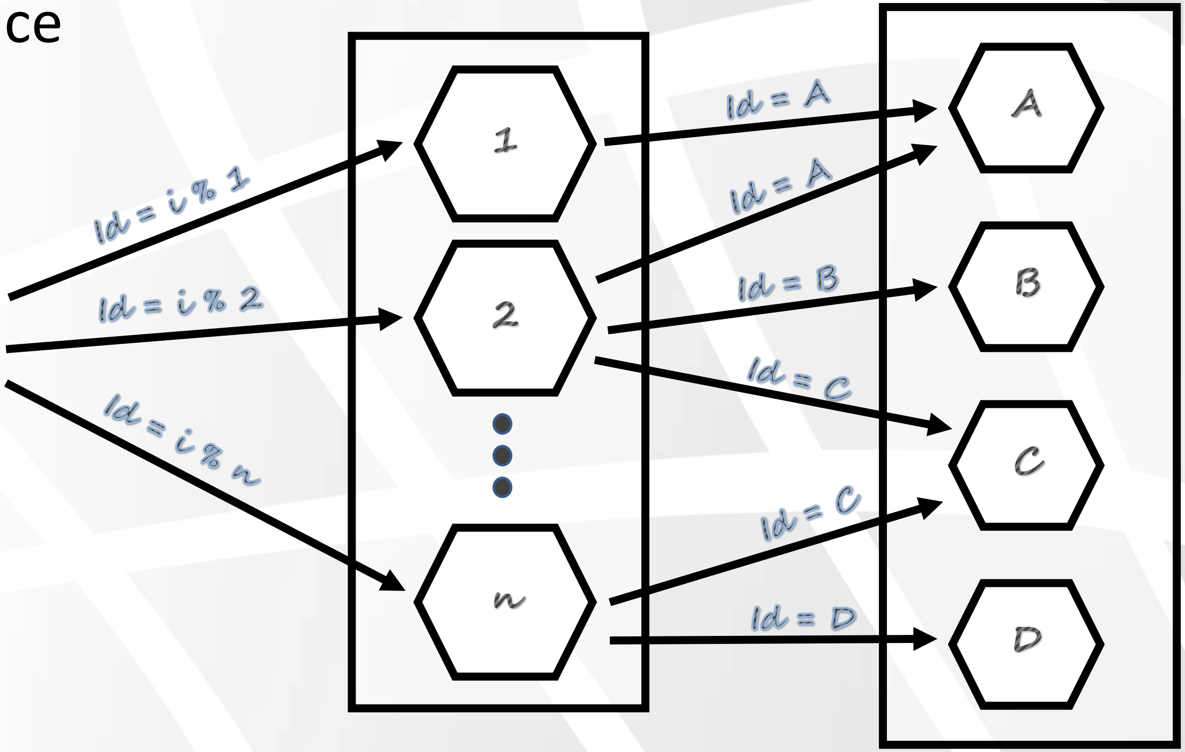
Id = Pattern

- Fork Join



Id = Pattern

- Map Reduce



Summary

Object Oriented

In-Proc
Compile time

Component Oriented

In-Proc
Isolation of logic
Available via reference

SOA

Distributed
Isolation of logic
Contract
Endpoints

Service Fabric (micro-services)

Distributed
Same as SOA +

- Distributed **IoC + GC**
- **Stateful** (and Stateless)
- **Dynamic Deployment**