



# Apache Ignite

## In-Memory Data Fabric for .NET

Dmitriy Setrakyan  
GridGain Founder & Chief Product Officer  
Apache Ignite PMC



[www.ignite.apache.org](http://www.ignite.apache.org)



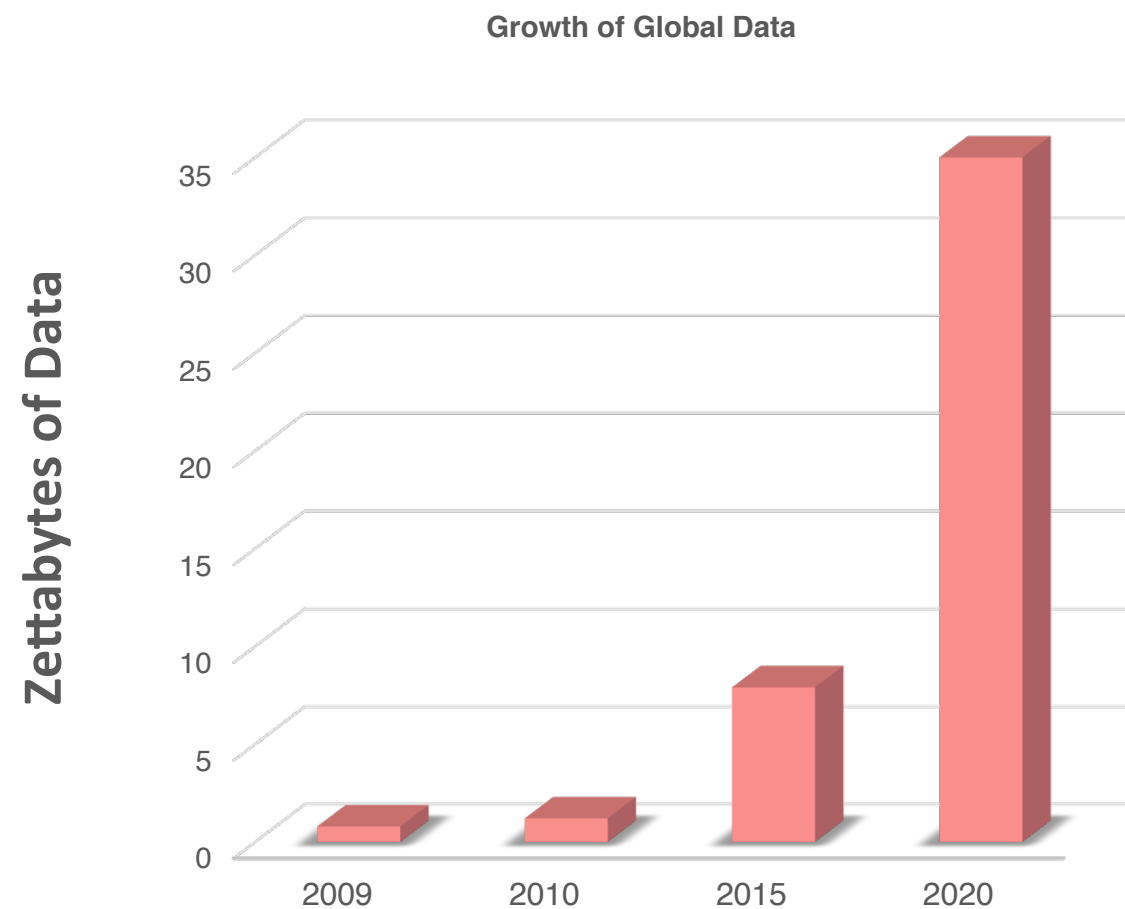
#apacheignite

# Agenda

- Project history
- **In-Memory Data Fabric**
  - In-Memory Clustering
  - In-Memory Compute Grid
  - In-Memory Data Grid
  - In-Memory Service Grid
  - In-Memory Streaming & CEP
  - Hadoop & Spark Accelerator

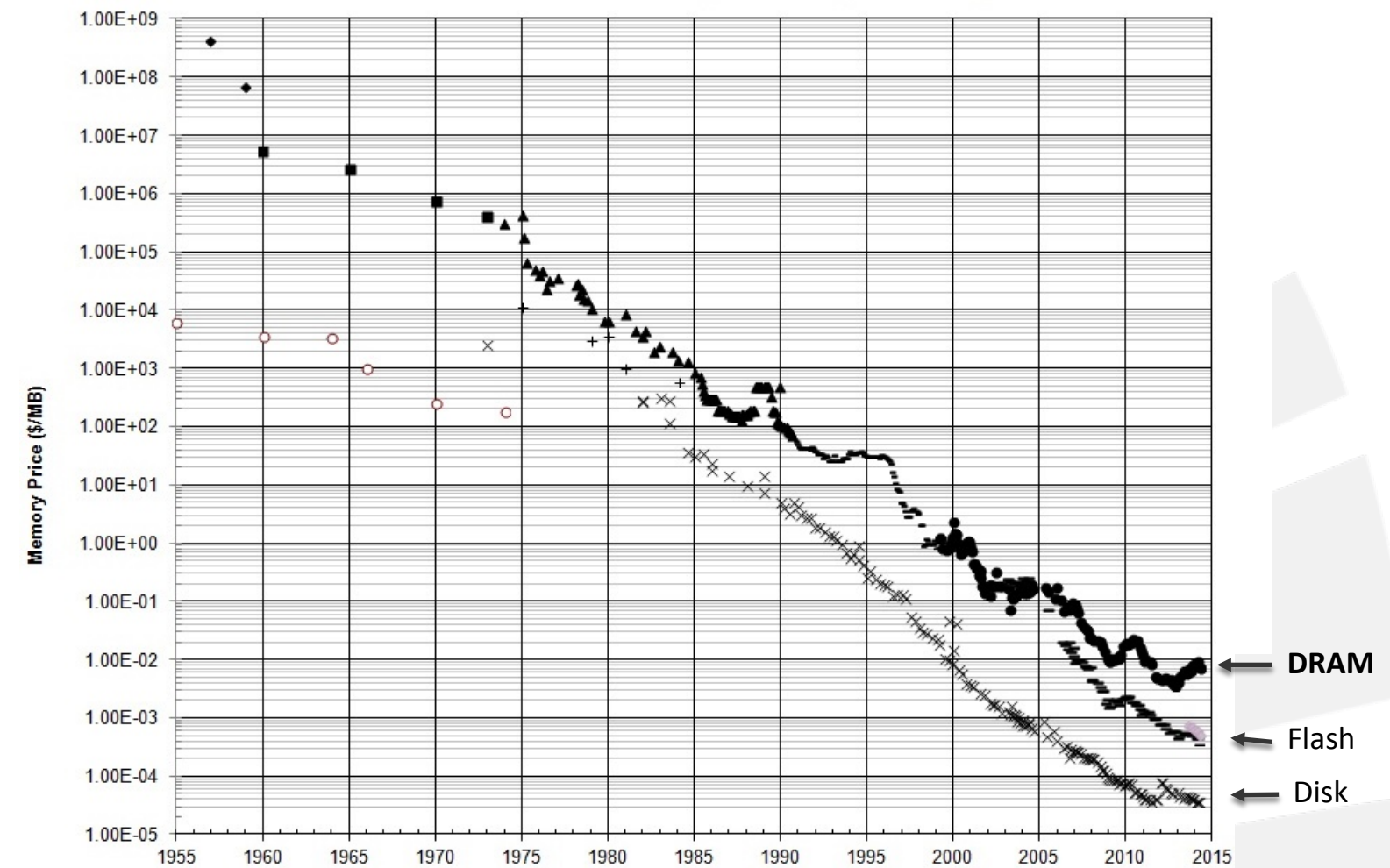
# Why In-Memory Computing Now?

## Data Growth Driving Demand



8 zettabytes in 2015 growing to 35 in 2020

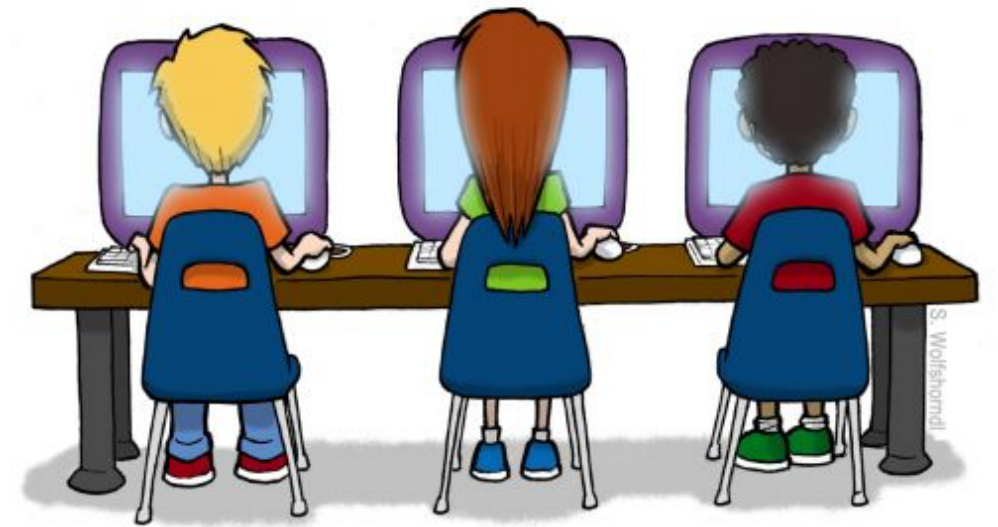
## Declining DRAM Cost



Cost drops 30% every 12 months

# Apache Ignite - We Are Hiring!

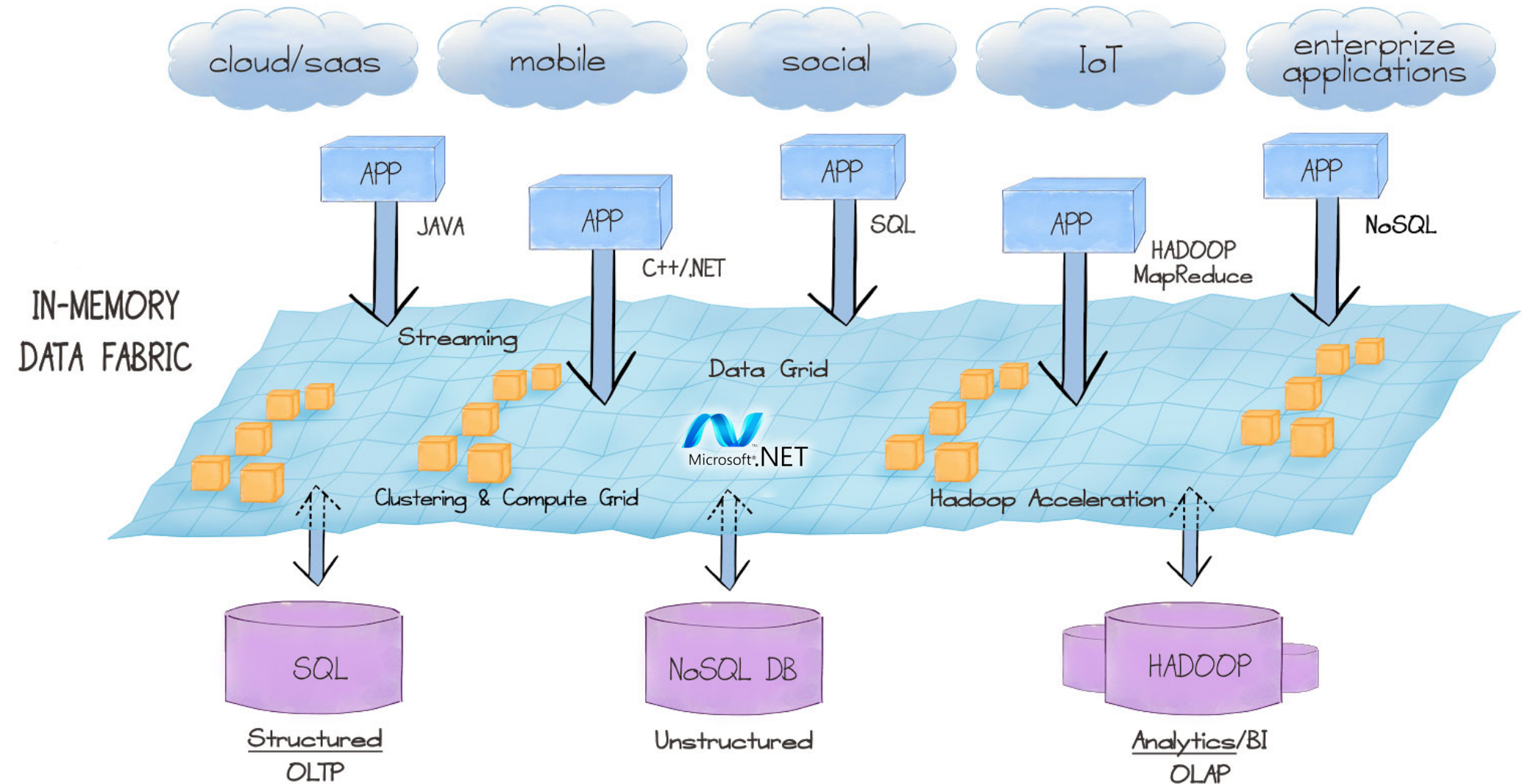
- Very Active Community
- Great Way to Learn Distributed Computing
- How To Contribute:
  - <https://ignite.apache.org/community/contribute.html#contribute>
  - <https://cwiki.apache.org/confluence/display/IGNITE/How+to+Contribute>



# In-Memory Data Fabric

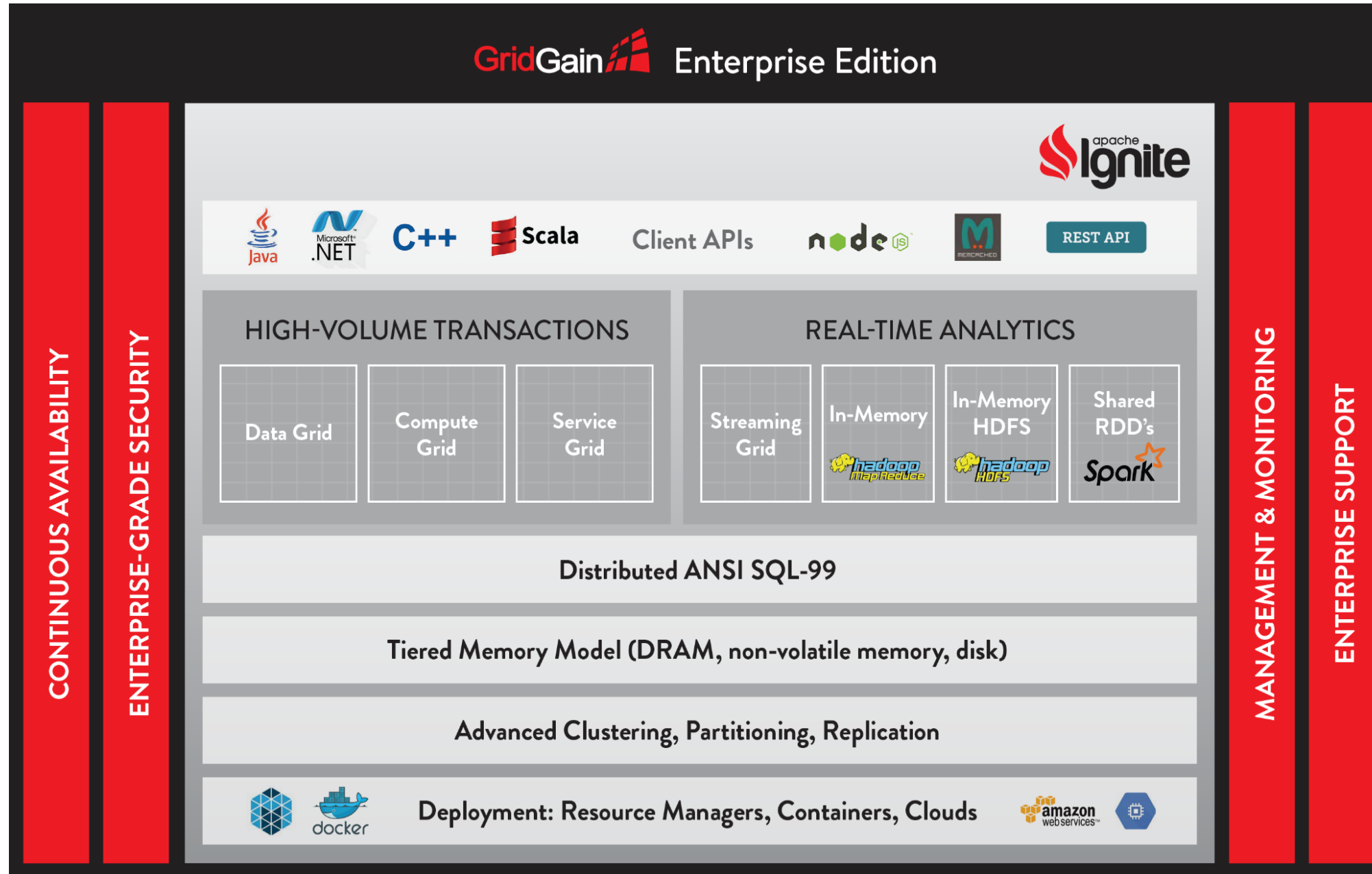
Natively developed for .NET

**Apache Ignite** is a leading open-source, cloud-ready distributed software delivering 100x performance and scalability by storing and processing data in memory across scale out or scale up infrastructure.





# Comprehensive In-Memory Data Fabric



- Richest functionality – a true data fabric
  - SQL, distributed data and compute grids, streaming, Hadoop acceleration, ACID transactions, etc.
- Open source
  - Apache Ignite – for Fast Data what Hadoop is for Big Data
- Most strategic, least disruptive approach to in-memory speed and scale
  - No rip-and-replace of existing infrastructure

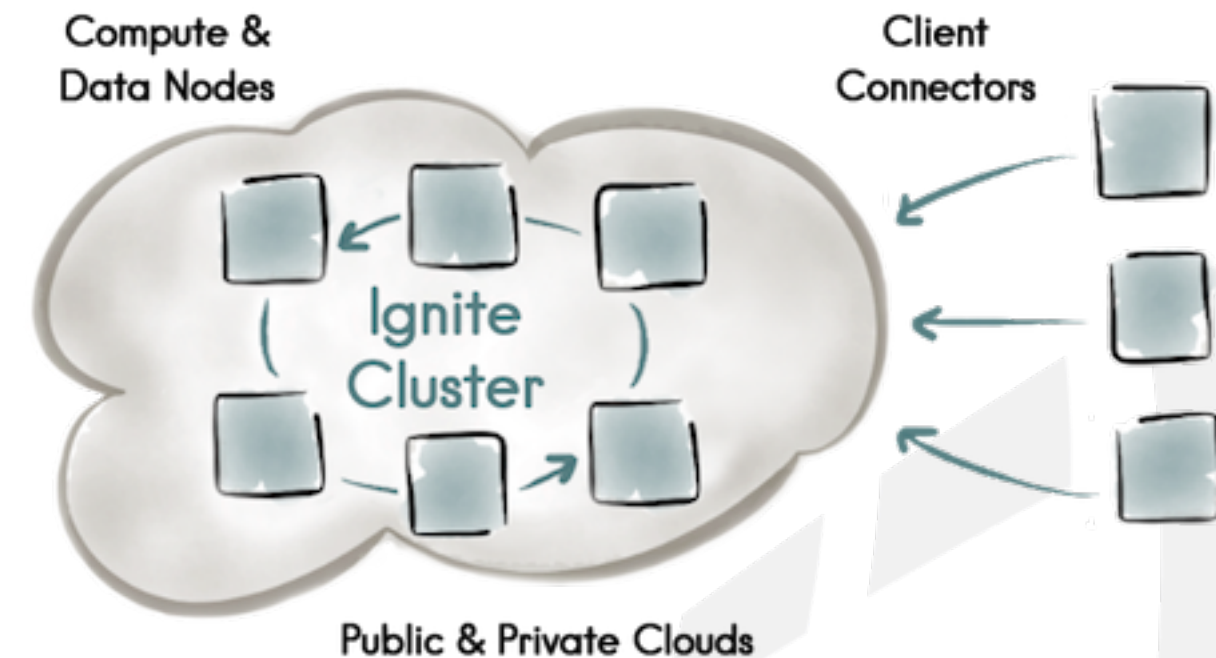
# Apache Ignite for .NET

- Built natively in .NET
- Built on top of Apache Ignite Java-core
  - Starts JVM inside of .NET process
- Binary serialization protocol
  - Read/write any data from/to any platform (Java, .NET or C++)
- 99% **native API** parity between Java and .NET
  - APIs, configuration, runtime are pure .NET
- Full LINQ integration:
  - Strongly typed, compiler-checked queries
- Full Java/.NET interop
  - Heterogeneous cluster
  - Java compute jobs can run on any Java, C++ or .NET node
  - .NET compute jobs can run only on .NET
- Hosted at NuGet



# In-Memory Clustering & Deployment

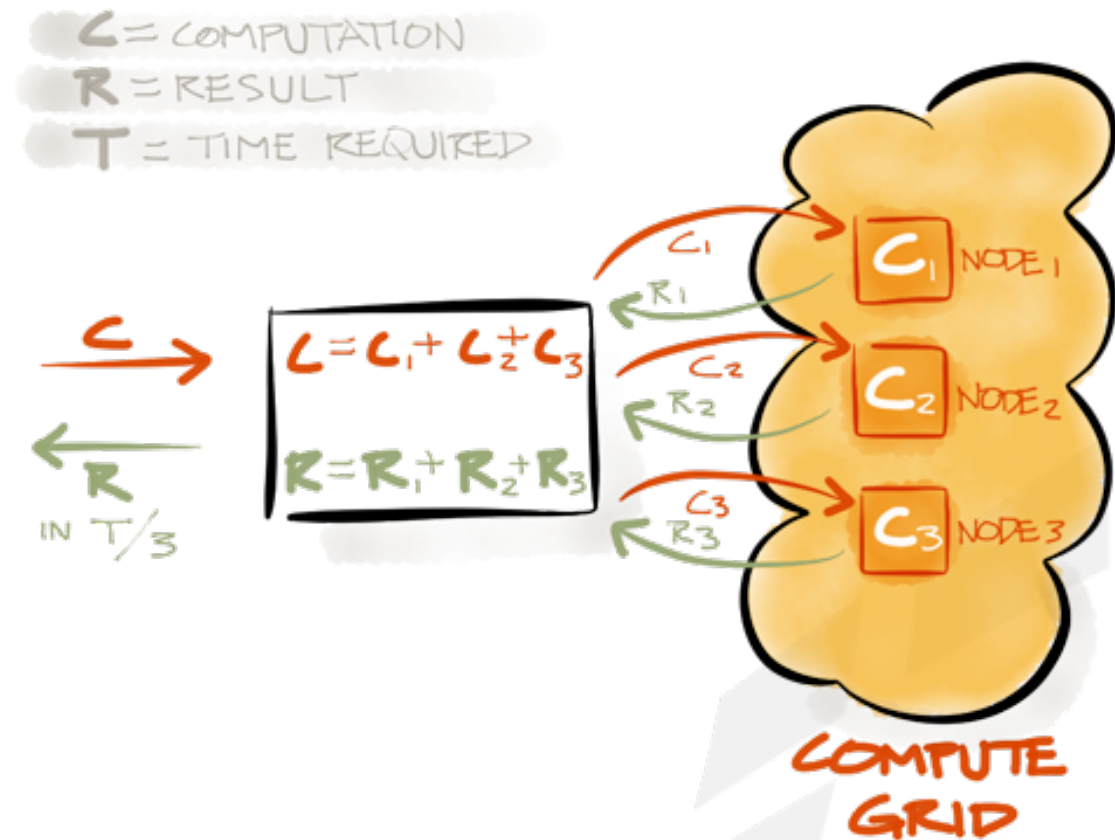
- Ease of Getting Started
  - Automatic Discovery
- Any Environment
  - Public Cloud
  - Private Cloud
  - Hybrid Cloud
  - Local Laptop
- Full Cluster Management
- Pluggable Design
- Available on Azure Marketplace





# In-Memory Compute Grid

- Direct API for MapReduce
- Zero Deployment
- Cron-like Task Scheduling
- State Checkpoints
- Load Balancing
- Automatic Failover
- Full Cluster Management
- Pluggable SPI Design





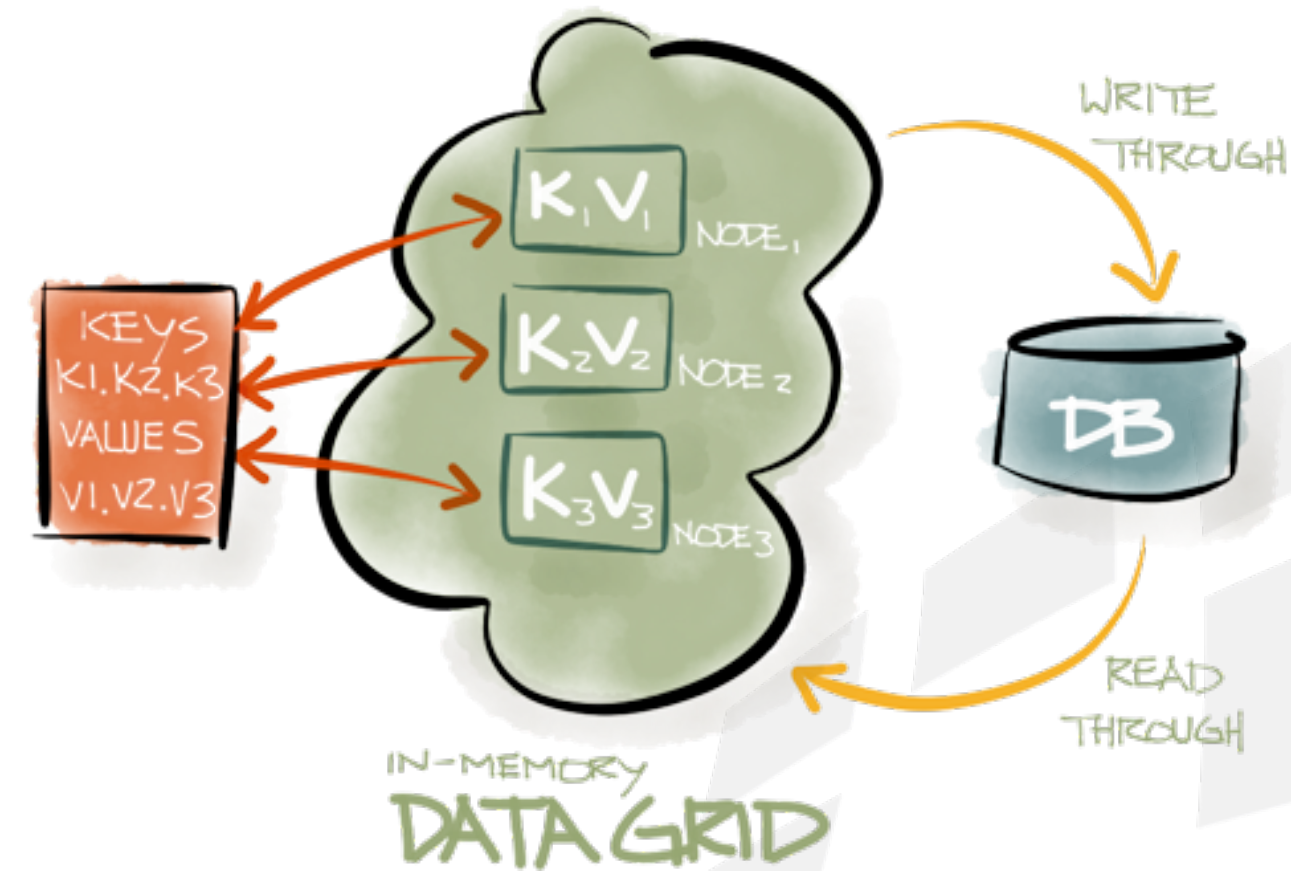
# Code Example

```
var words = "Count characters using closure".Split();  
var res = ignite.GetCompute().Apply(  
    new CharacterCountClosure(), words  
);  
Console.WriteLine(  
    "Total character count with manual reduce: ", res.Sum()  
);
```

```
[Serializable]  
public class CharacterCountClosure : IComputeFunc<string, int>  
{  
    public int Invoke(string arg) => arg.Length;  
}
```

# In-Memory Data Grid

- Distributed In-Memory Key-Value Store
- Replicated and Partitioned data
- TBs of data, of any type
- On-Heap and Off-Heap Storage
- Highly Available In-Memory Replicas
- Automatic Failover
- Distributed ACID Transactions
- SQL99 queries and JDBC/ODBC driver
- Collocation of Compute and Data





# Code Example

```
ICache<int, Organization> cache =  
    ignite.CreateCache<int, Organization>("orgs");  
cache.Put(1, new Organization {Name = "Apache", Type="Private"});  
Console.WriteLine(  
    "Retrieved organization instance from cache: ",  
    cache.Get(1)  
);
```

```
ICache<int, IBinaryObject> binaryCache =  
    cache.WithKeepBinary<int, IBinaryObject>();  
IBinaryObject binaryOrg = binaryCache.Get(1);  
Console.WriteLine(  
    "Retrieved organization name from binary object: ",  
    binaryOrg.GetField<string>("name")  
);
```



# Code Example

```
const int zip = 94109;
```

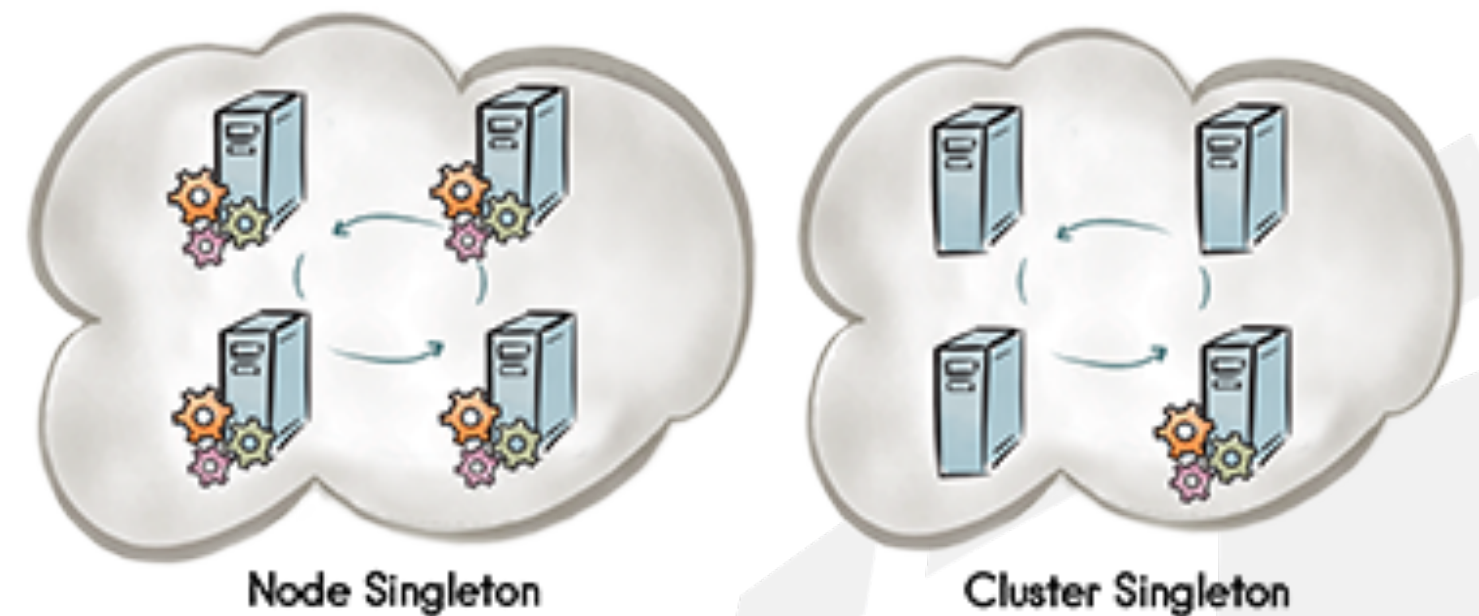
```
IQueryable<ICacheEntry<EmployeeKey, Employee>> qry =  
    cache.AsCacheQueryable().Where(emp => emp.Value.Address.Zip == zip);
```

```
Console.WriteLine();
```

```
Console.WriteLine("Employees with zipcode " + zip + ":");  
foreach (ICacheEntry<EmployeeKey, Employee> entry in qry)  
    Console.WriteLine(entry.Value);
```

# In-Memory Service Grid

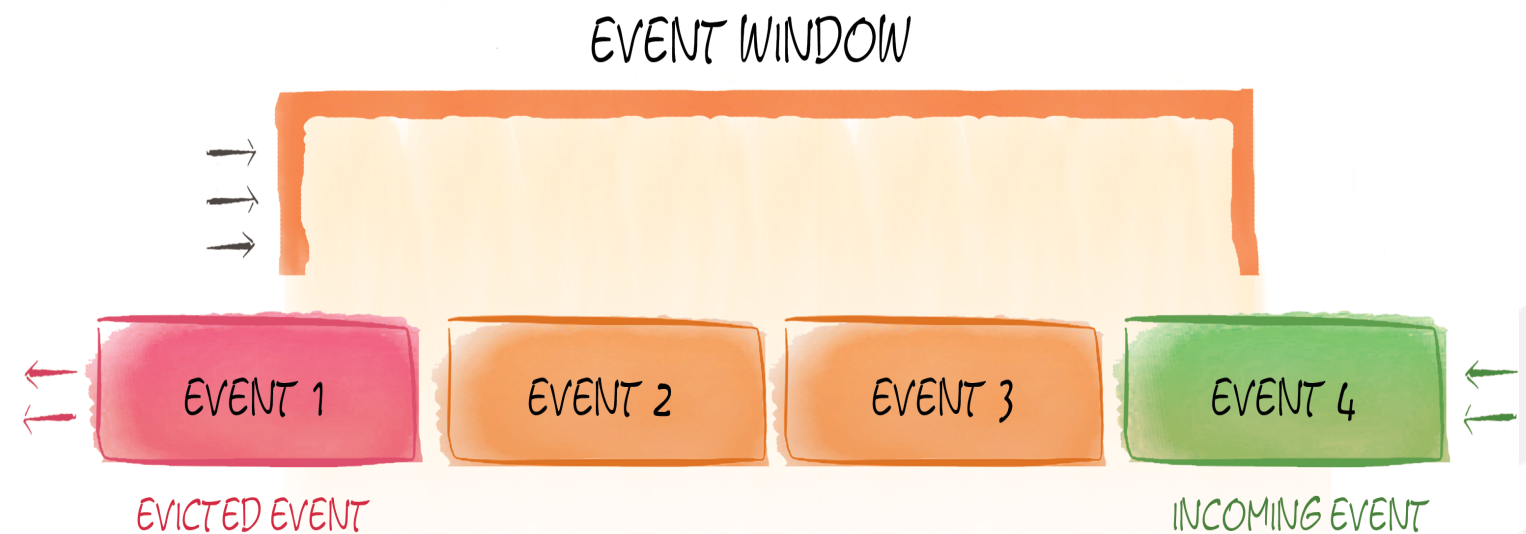
- Distribute Any Data Structure
  - Available Anywhere on the Grid
  - Automatic Remote Access via Proxies
- Controlled Deployment
  - Support for Cluster Singleton
  - Support for Node Singleton
  - Support for Custom Topology
  - Load Balanced
- Guaranteed Availability
  - Auto Redeployment in Case of Failures





# In-Memory Streaming and CEP

- Streaming Data Never Ends
- Branching Pipelines
- Pluggable Routing
- Sliding Windows for CEP/Continuous Query
- Real Time Analysis

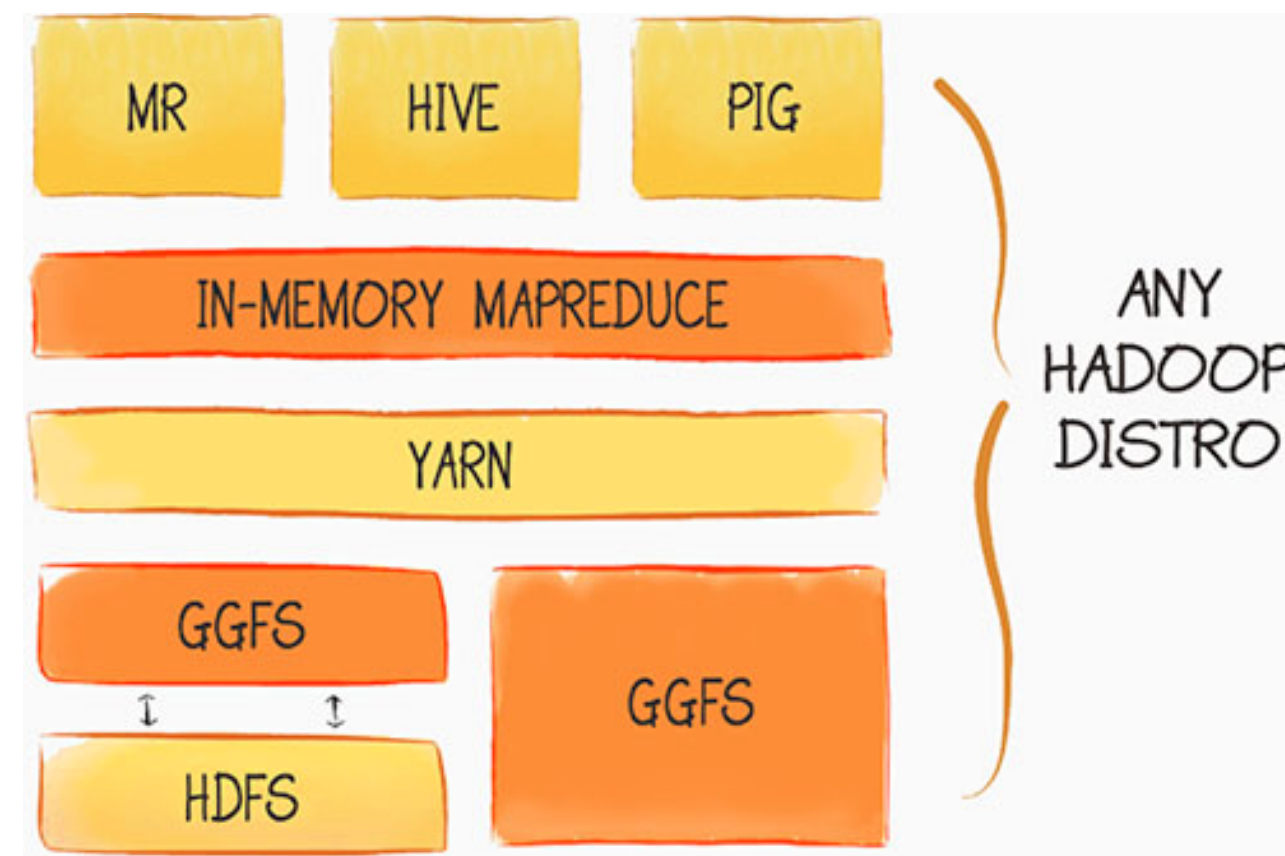


# .NET Code Example

```
using (IDataStreamer<int, Account> ldr =  
    ignite.GetDataStreamer<int, Account>("accounts"))  
{  
    for (var i = 0; i < 500000; i++)  
        ldr.AddData(i, new Account(i, i));  
}
```

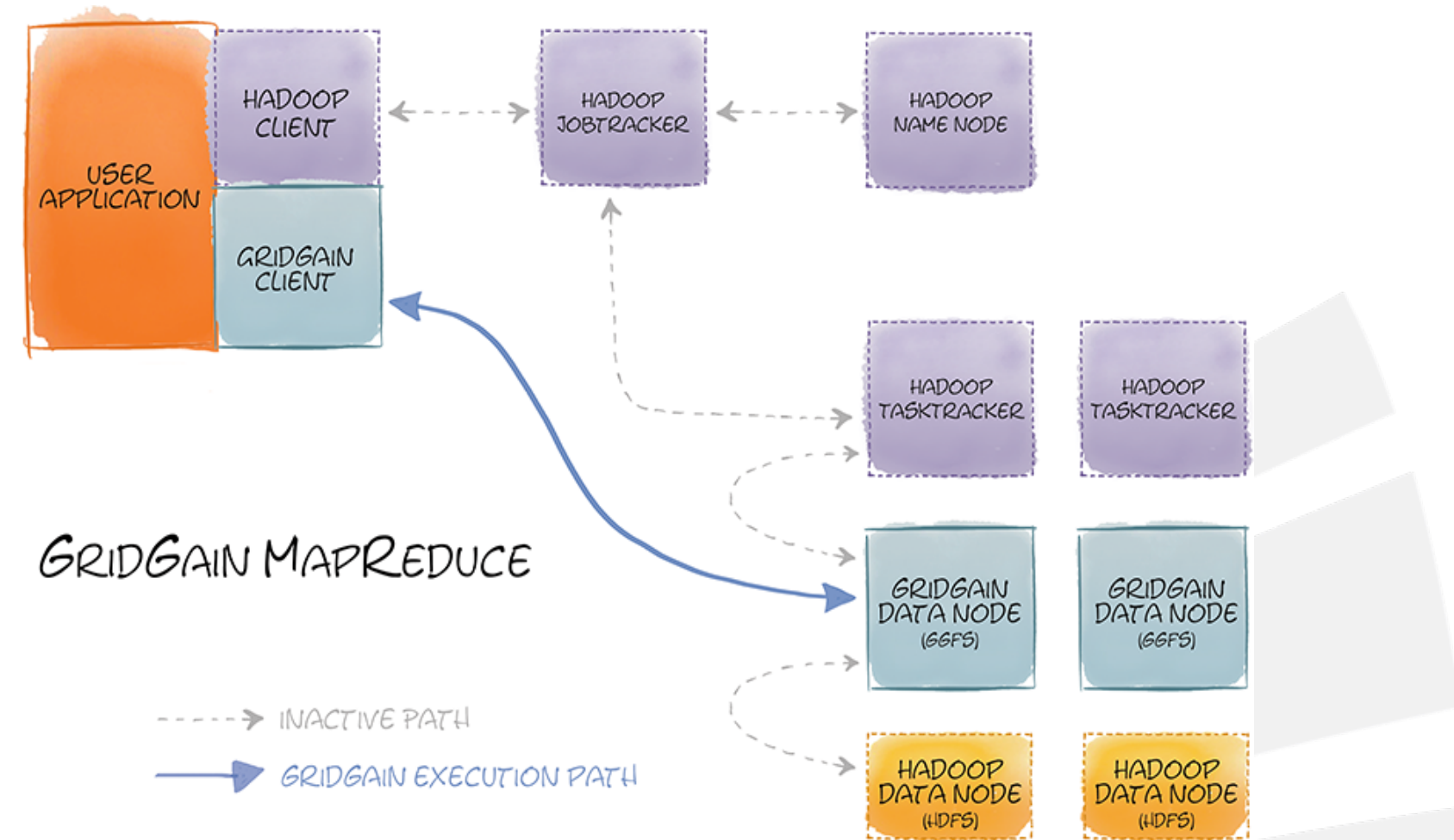
# In-Memory Hadoop Accelerator

- **Plug and Play installation**
- 10x to 100x Acceleration
- In-Memory Native MapReduce
- In-Process Data Colocation
- IgniteFS In-Memory File System
- Read-Through from HDFS
- Write-Through to HDFS
- Sync and Async Persistence



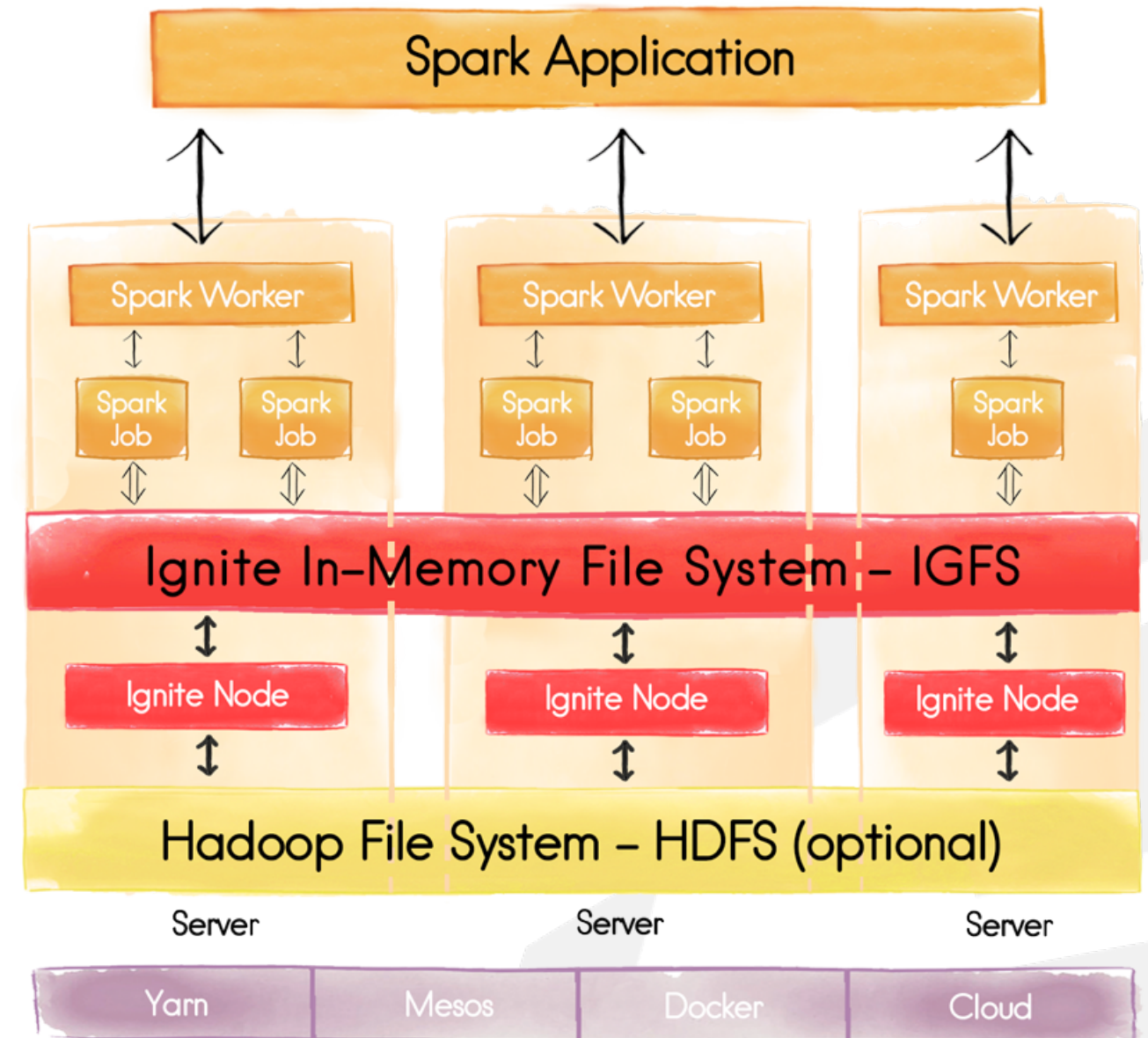
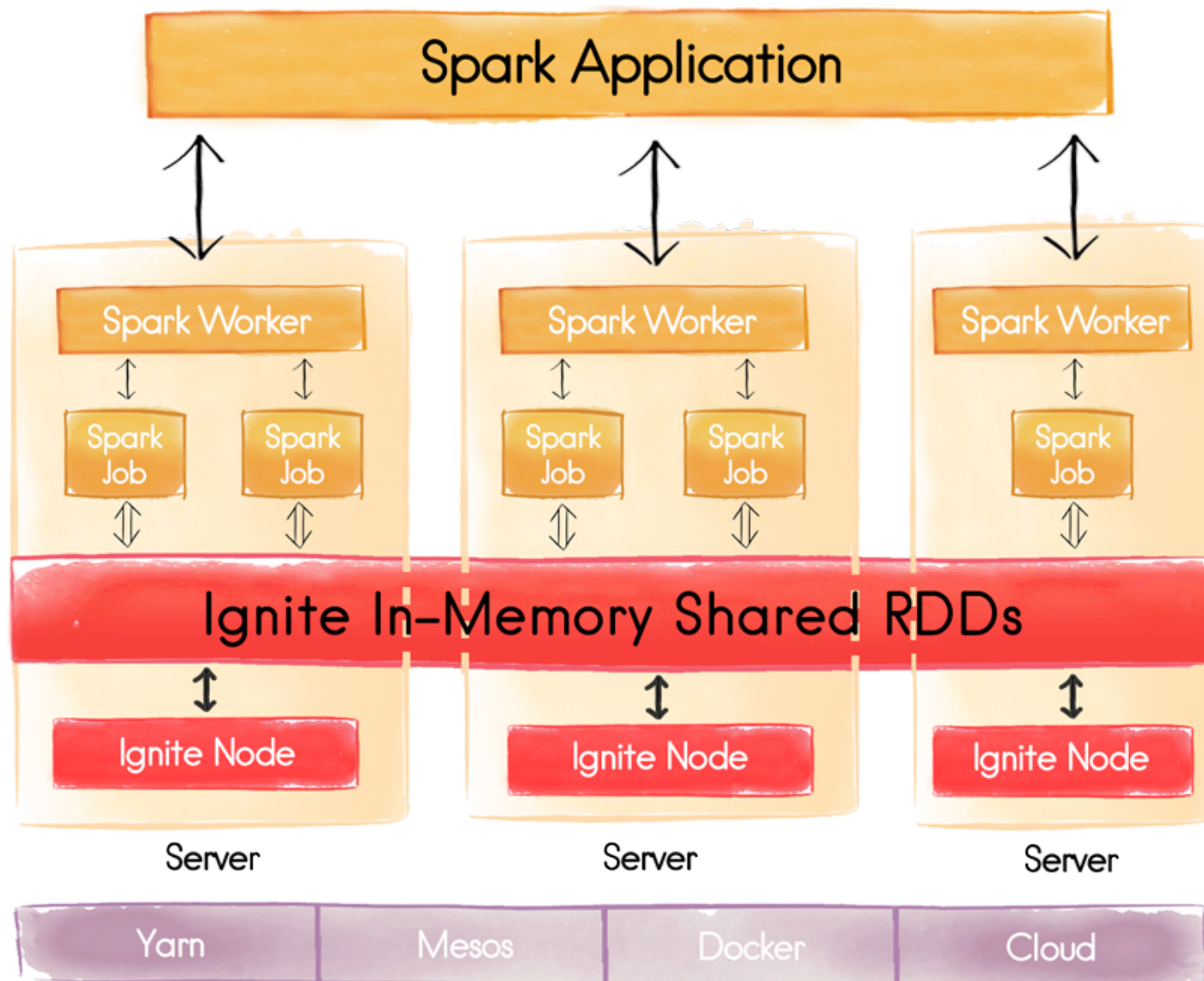
# In-Memory Hadoop Accelerator

- **Zero Code Change**
- **In-Memory Native Performance**
- Use existing MR code
- Use existing Pig/Hive queries
- No Name Node
- Eager Push Scheduling



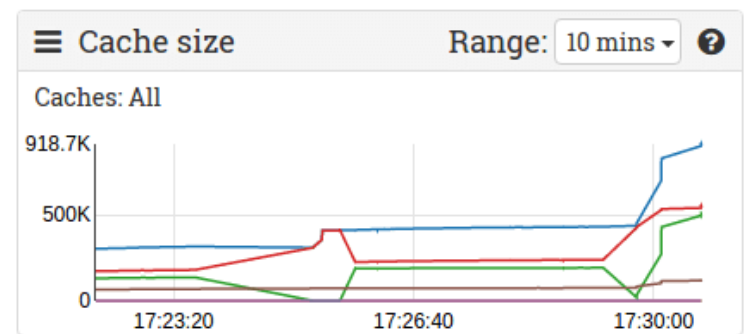
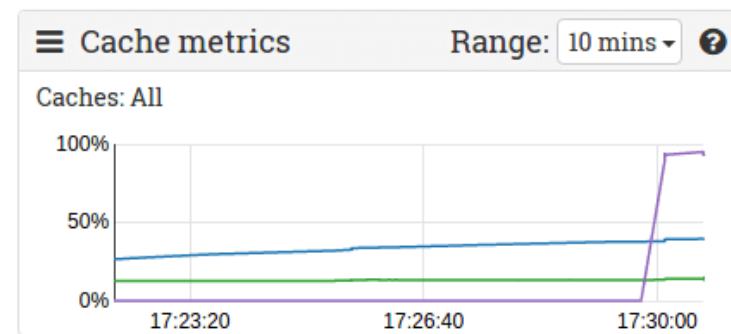
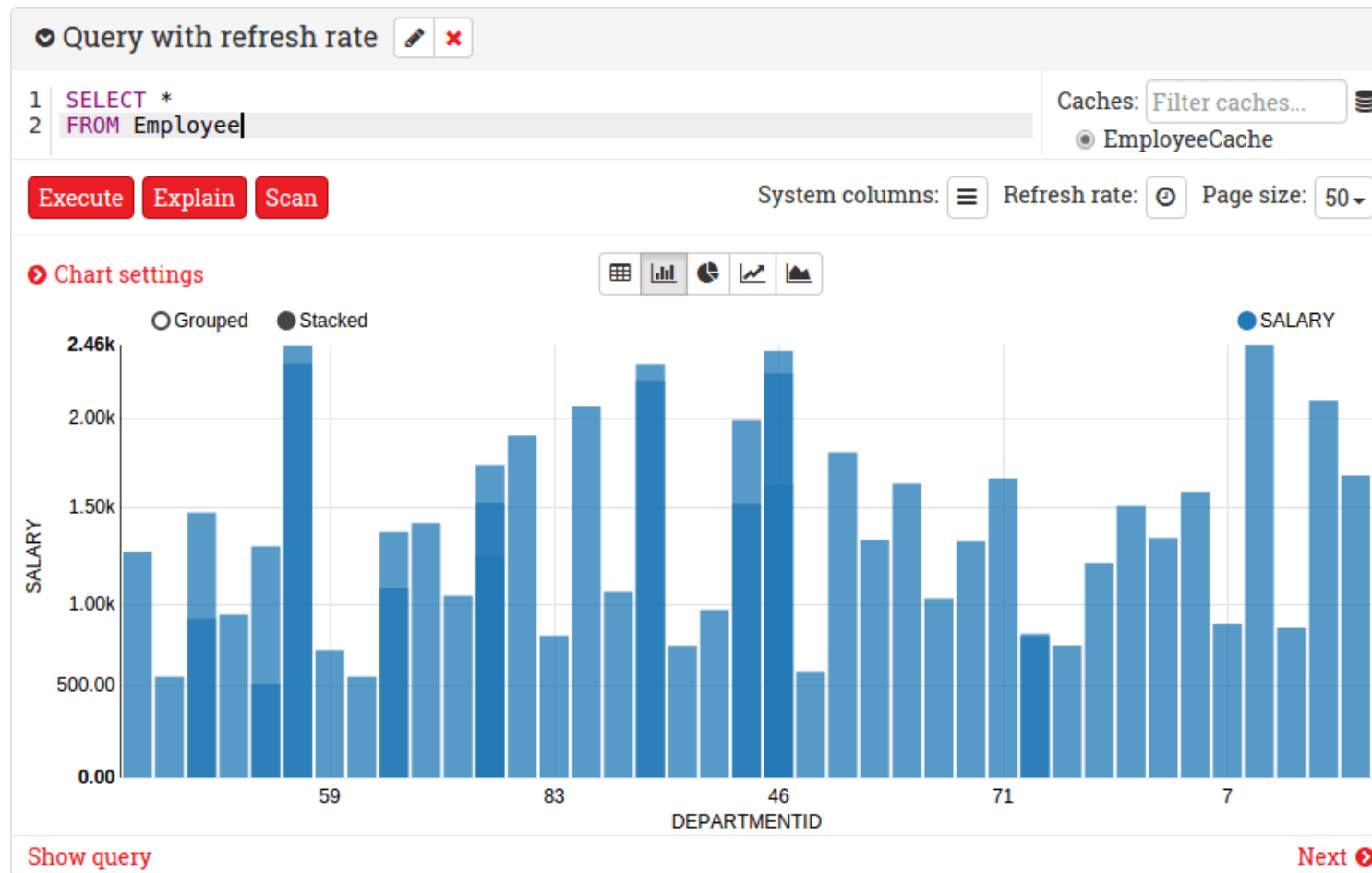







# Spark Integration – IGFS & Shared RDD











# GridGain Management Console

<https://console.gridgain.com>



Nodes: 2 Selected: 0all      

Node ID8	M	Node IP	Version	Start time	Up time	Busy	CPU%	CPU load	GC load	Fre
EC0B10AA	S	192.168.1.20	7.6.0-SNAPS...	Jun 07, 17:15:41	7m 56s	0.85%	8	10.07%	0.27%	

**Caches: 19** Selected: 0all        

Name	M	Memory usage			On-heap entries				Off-heap entries		
		Heap	Off-heap	Swap	Total	Primary	Backup	Near	Total	Primary	Backup
c_local	L	16.2KB	0	0	77	77	0	0	0	0	
c_partitioned	P	184.8MB	0	0	600K	300K	300K	0	0	0	
c_partitioned2	P	4.8KB	374.6KB	0	20	10	10	0	2.4K	1.2K	1.2K





# THANK YOU!

<http://www.ignite.apache.org>



@apacheignite



@dsetrakyan

