Achieving Scalability and High Availability for clustered Web Services using Apache Synapse

Ruwan Linton [ruwan@apache.org]
WSO2 Inc.

Leading the Wave of Open Source

Contents

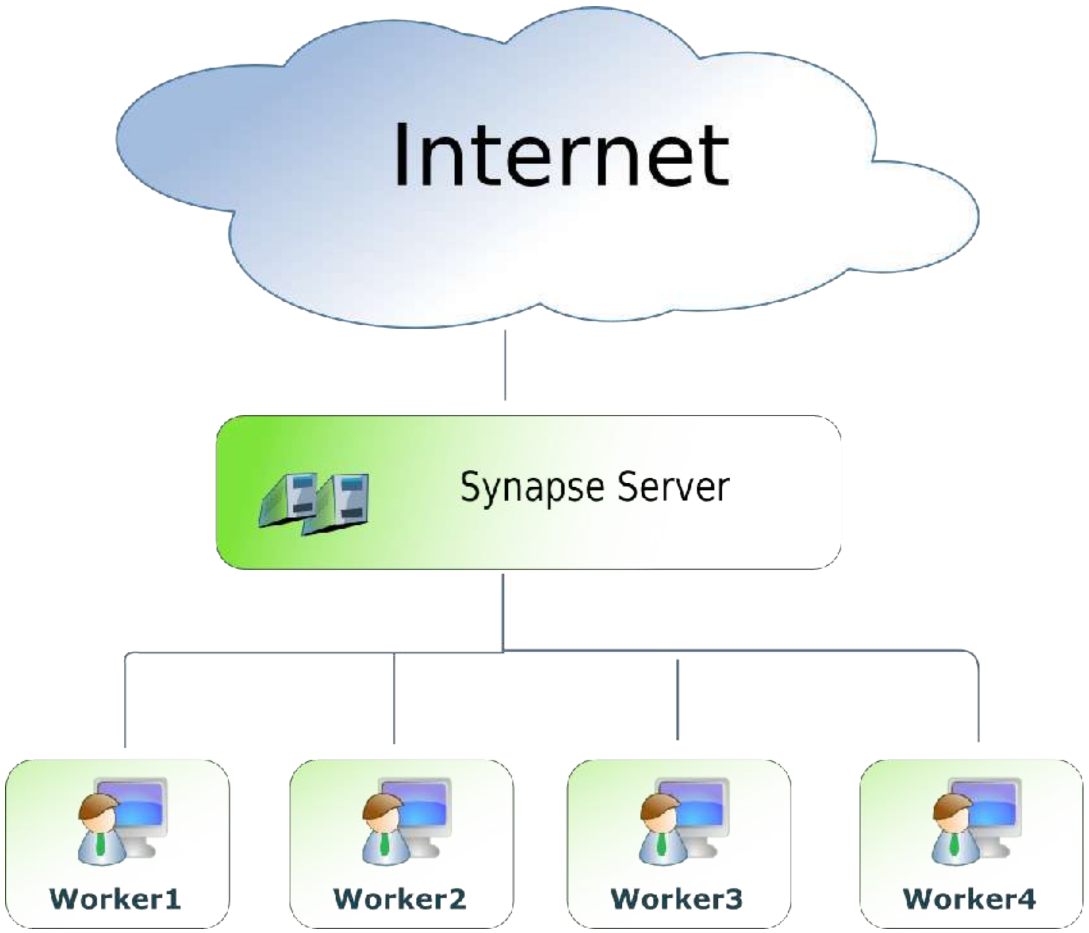
- Introduction
  - Apache Synapse
  - Web services clustering
  - Scalability/Availability in general
- Configuration
  - Proxy Service
  - Load balance and Fail over endpoints

Contents (Cntd..)

- Architecture
  - Stateless load balancing
  - State full load balancing
  - Load balancing with fail over
  - Dynamic load balancing
- Deployment

Apache Synapse

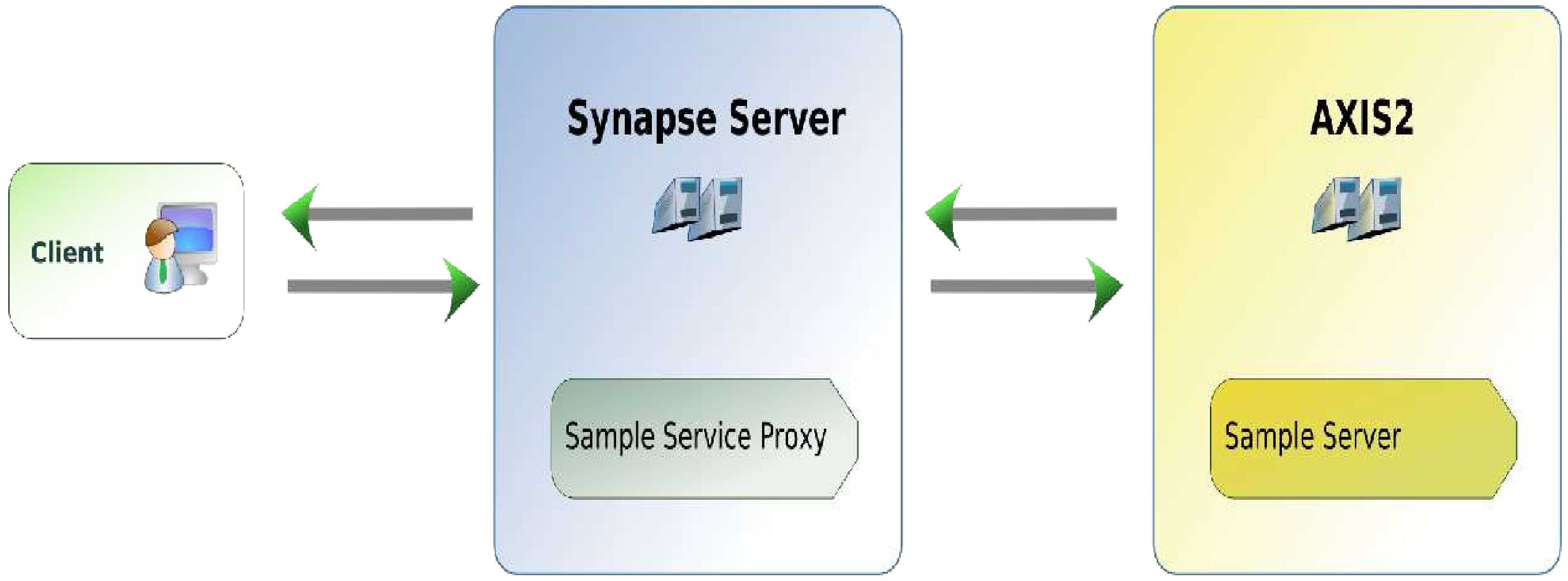
- Open source ESB providing mediation
- Configurable via an XML info set
- Lightweight, asynchronous core with streaming in the HTTP/S transports
- Extensible with scripts, or Java
- Can act as a front server for the web service infrastructure with load balancing, fail over and various mediation capabilities


Internet
Synapse Server
Worker1
Worker2
Worker3
Worker4

Synapse operational models

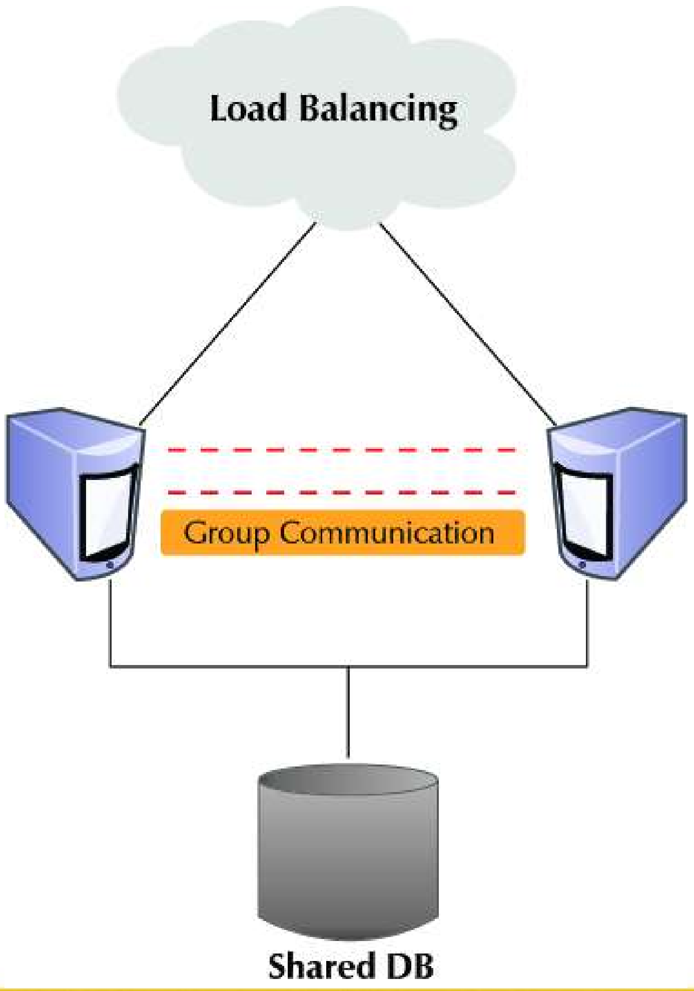
- Virtual Service (Proxy service) model
- Message mediation model
- Scheduled work
- Event driven architecture (Event broker)

Virtual Services



Web services clustering

- Different servers providing the same service
- Group communication within the servers
- Shared resources for the cluster



Scalability

Ability to either handle growing amounts of work in a graceful manner, or to be readily enlarged

Availability

The degree to which a system, subsystem, or equipment is operable and in a committable state at the start of a mission, when the mission is called for at an unknown

Synapse Configuration

- Configure a proxy service as the virtual service for the cluster of services
  - wrapped with a <proxy> element
  - declare an incoming/outgoing mediation
  - declare the endpoint
  - configure any quality of service improvements

Sample proxy

```
<proxy name=”foo”>
  <target inSequence=”ref-seq”>
    <endpoint>
      <address uri=”http://host/service”/>
    </endpoint>
  <target>
</proxy>
```

Synapse Config (Cntd..)

- Endpoint configuration of the proxy
  - 5 types with 3 primitive and 2 secondary
  - wrapped with an <endpoint> element
  - declare the actual endpoint properties
  - and any optimizations

Sample LB endpoint

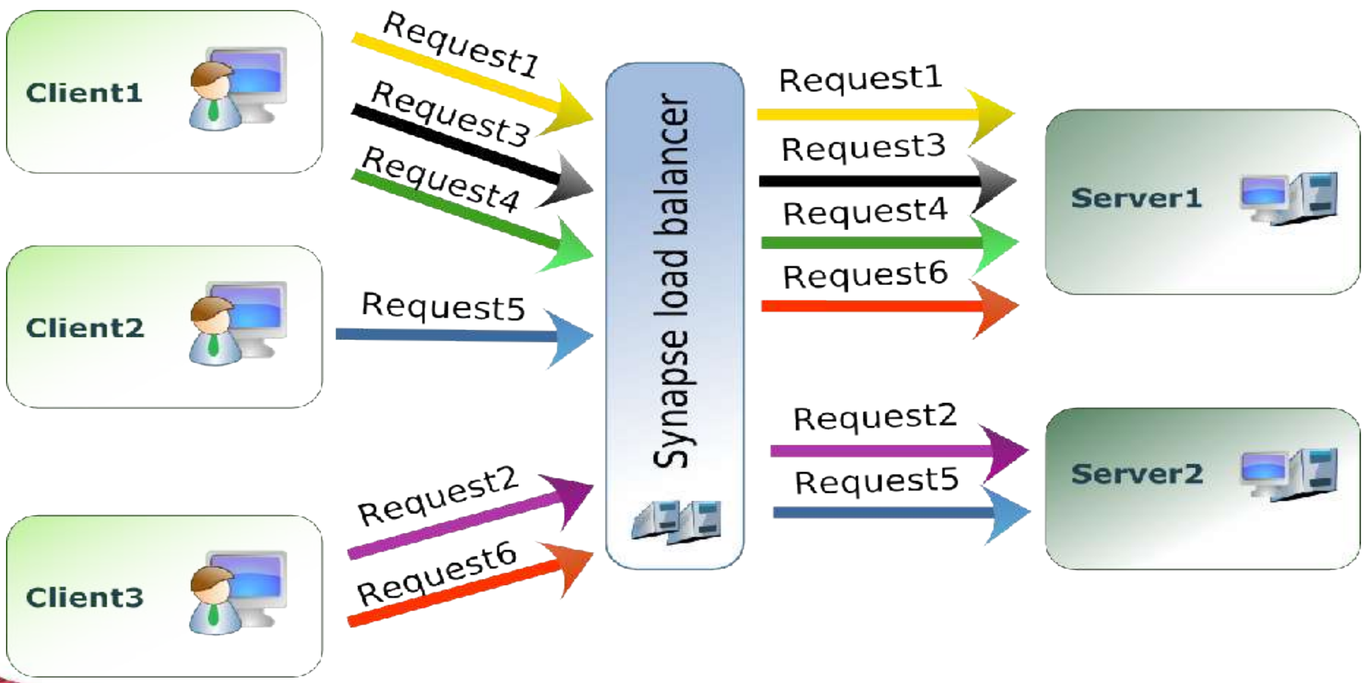
```
<endpoint name="test-lb">
    <session type="soap"/>
    <loadbalance policy="roundRobin">
        <endpoint ..../>
        ..
        ..
    </loadbalance>
</endpoint>
```

Architecture

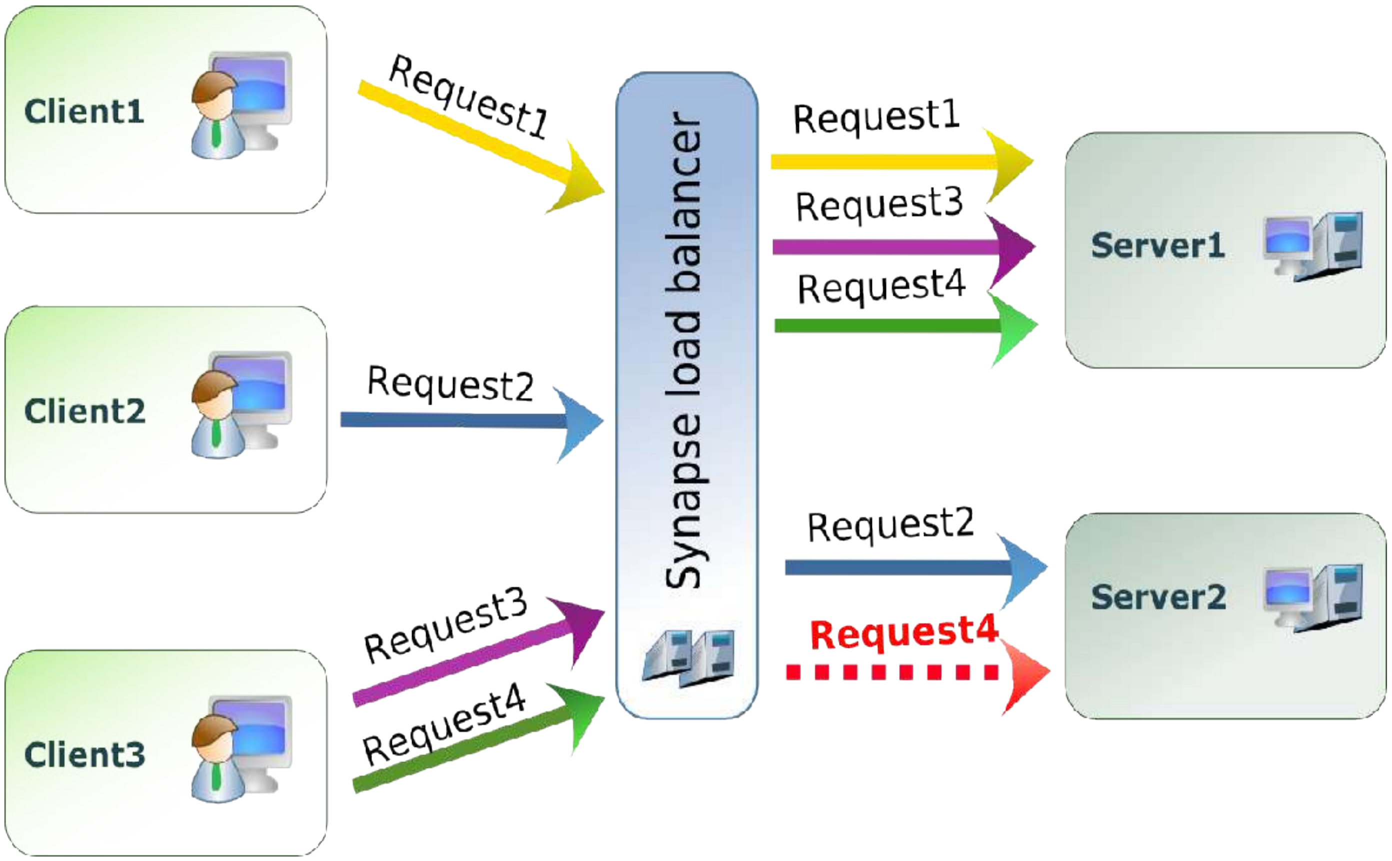
- Stateless load balancing
  - LB algorithm implementation is pluggable
  - Built in round robin algorithm
  - Weighted LB algorithm

State full session aware load balancing

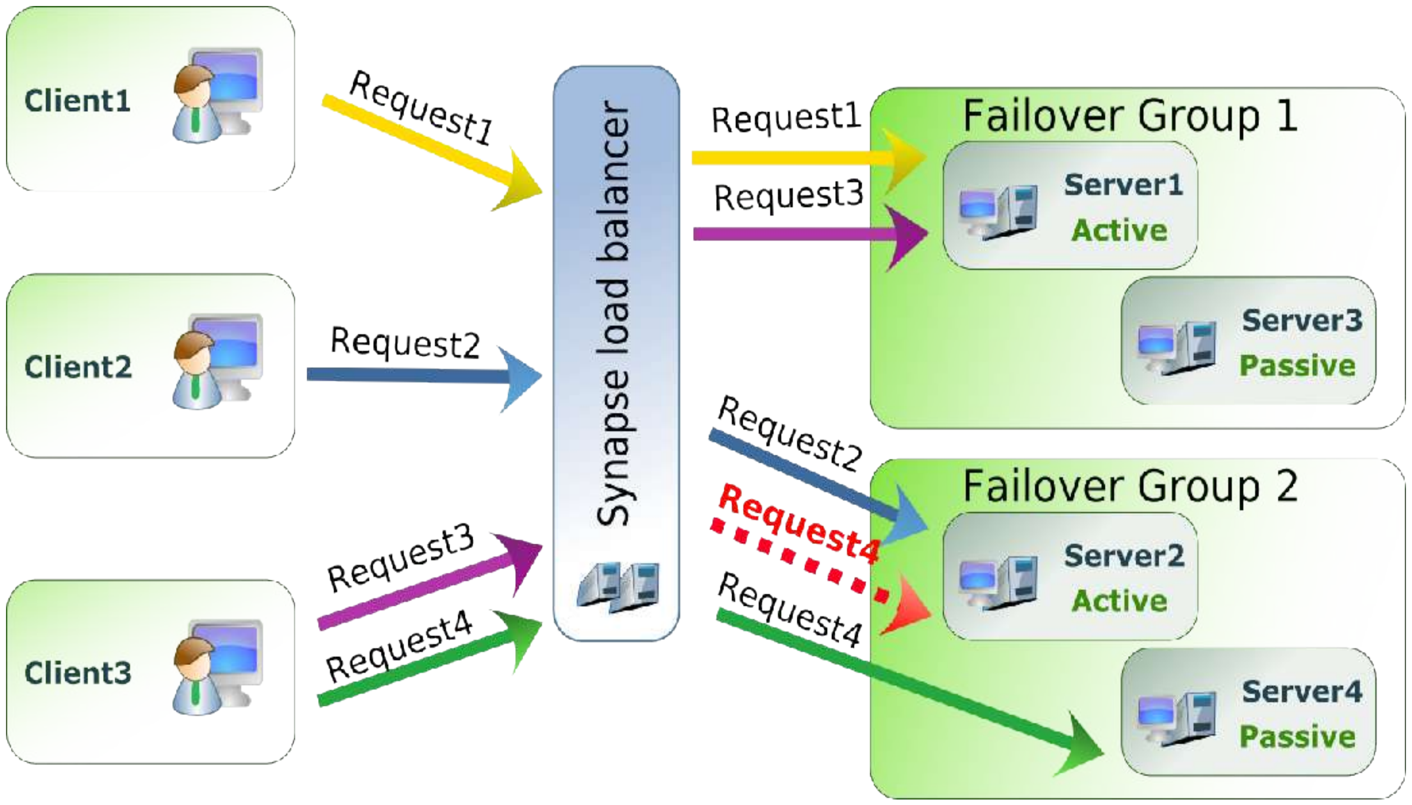
- Transport and SOAP session
- Client initiated vs Server initiated



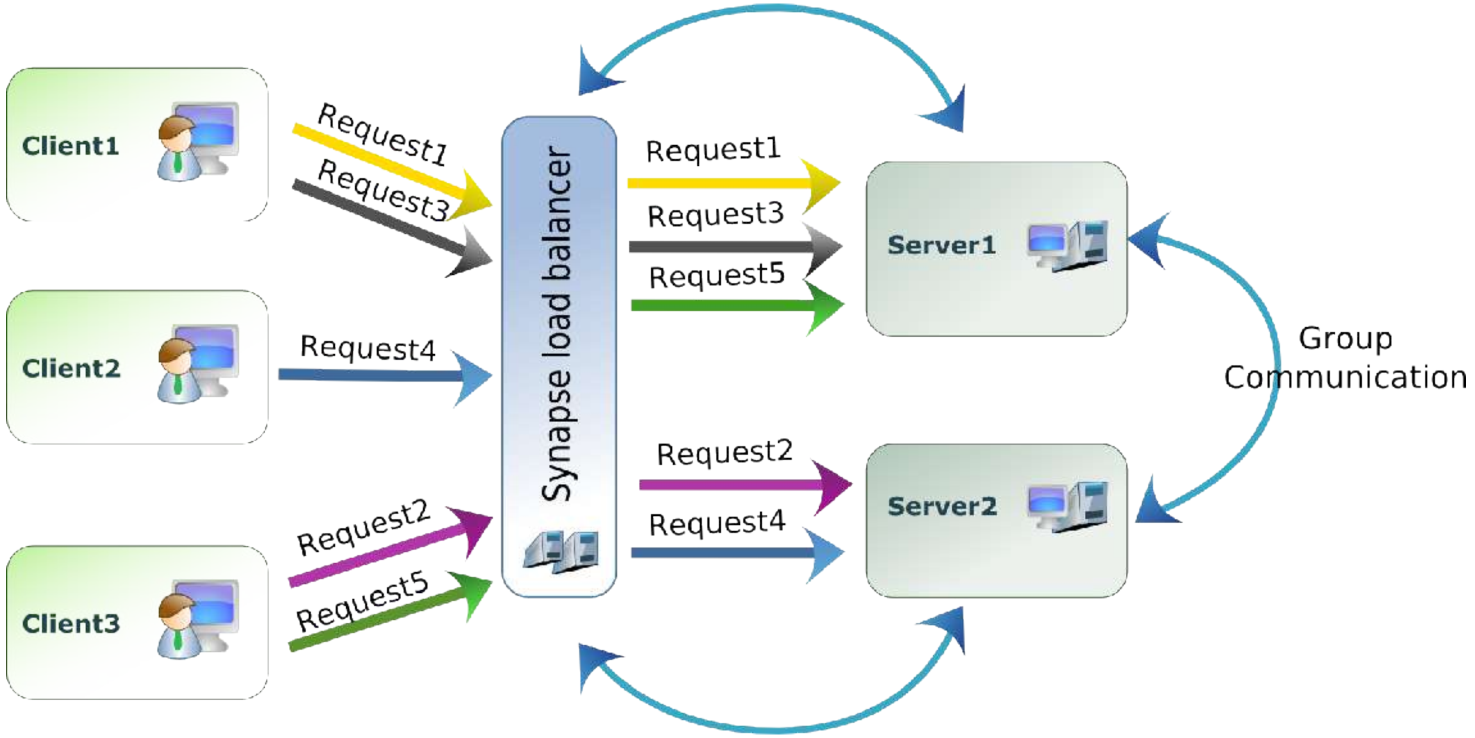
Fail over with LB



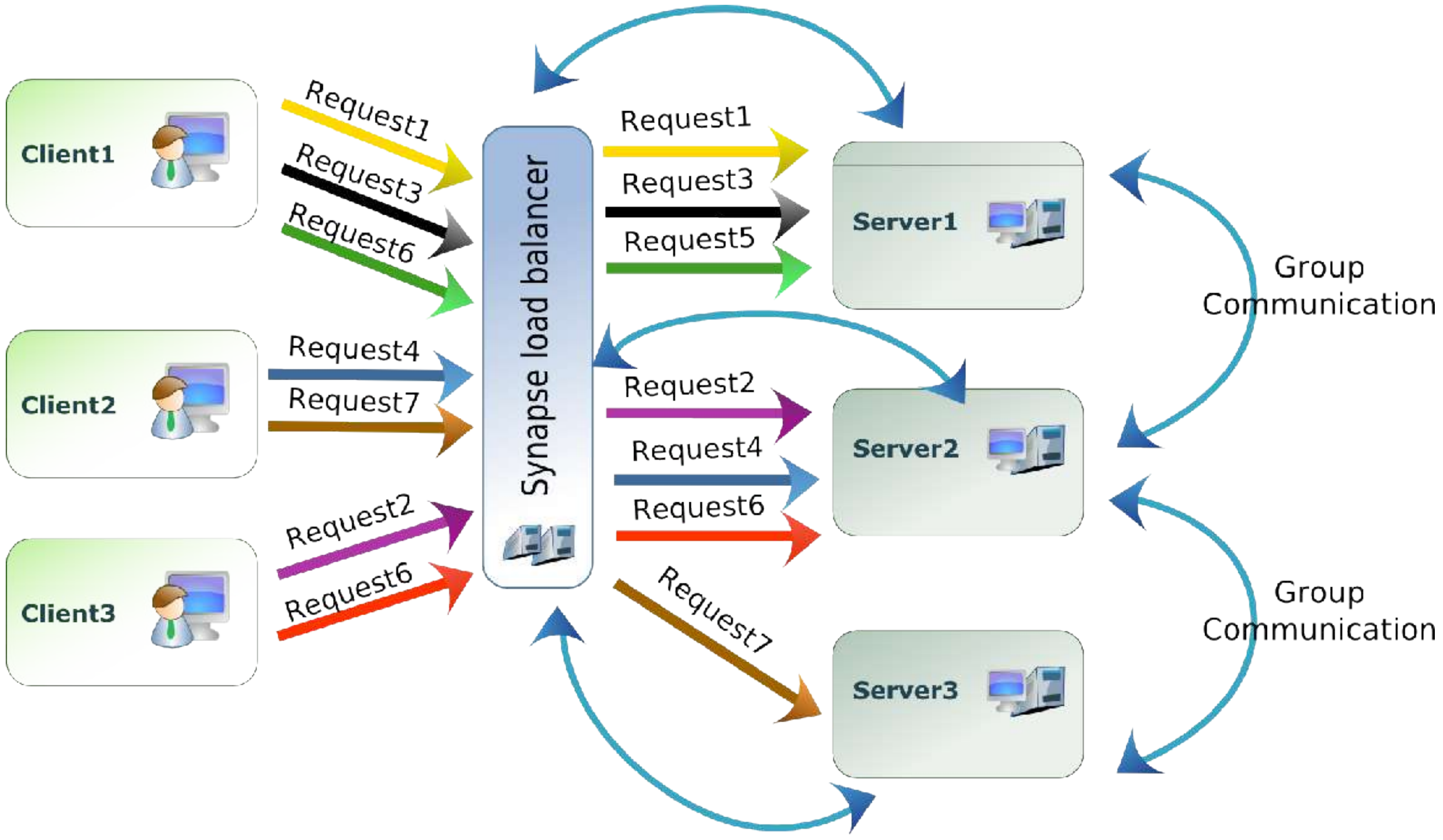
LB and FO groups



Dynamic LB


Client1
Client2
Client3
Request1
Request3
Request4
Request2
Request5
Synapse load balancer
Request1
Request3
Request5
Request2
Request4
Server1
Server2
Group Communication

Dynamic LB (Cntd..)



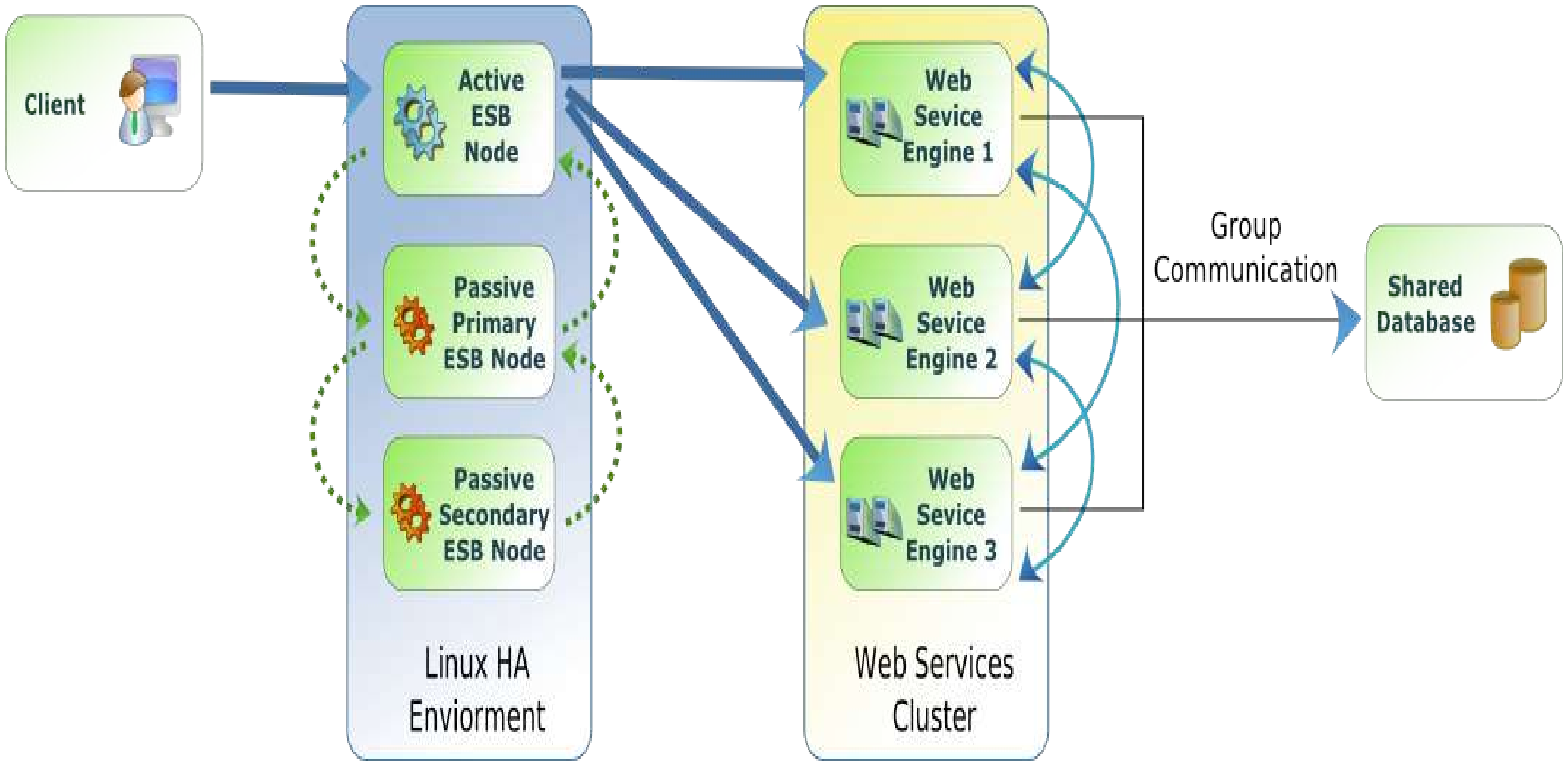
Scalability of Synapse

- Can handle 2500 concurrent connections
- Can handle 30M transactions per day
  - assumption: one transaction is one request for Synapse
- Non-blocking HTTP/S transport with message queuing with a configurable thread pools
- Different thread pool at the application layer and the I/O layer

Availability of Synapse

- Availability can be achieved with the deployment
  - Two passive nodes with a given active node in vertically
- Graceful shutdown and maintenance mode
  - Shutdown the listeners
  - Let the senders send out the responses after processing the already accepted messages
  - Shutdown the senders and the server or upgrade and restart the listener manager
- Round robin restart of the cluster in active active deployment

Deployment diagram



Analogy

- Isn't it just shifting the scalability and availability to synapse layer?
  - Yes it is... but if you look at the availability and scalability of a typical web service hosting environment and synapse with the non blocking HTTP/S it is much scalable and available than the service hosting environment
  - Fail over and Load balancing ability of synapse increases the availability and scalability of web services in the cluster

Analogy (Cntd..)

- Once you have the auto scaling implemented the dynamic load balancing functionality fits with this nicely to achieve the scalability into great extent
- You get many other features with Synapse as the load balancer like session aware load balancing at the SOAP session

More..

- Throttling and caching at the Synapse layer
  - Concurrency throttling handles the scalability gracefully by rejecting the messages
  - Caching improves the availability
- Fault tolerance through a fault handling mechanism

Summary

- Synapse load balancing and fail over routing can be used to achieve high availability of the web services
- Synapse is scalable with the non locking transport and streaming
- Session aware load balancing on SOAP message is an added advantage of using synapse
- Availability of the web services can be achieved with the the correct deployment with synapse

Thank You!

Questions??