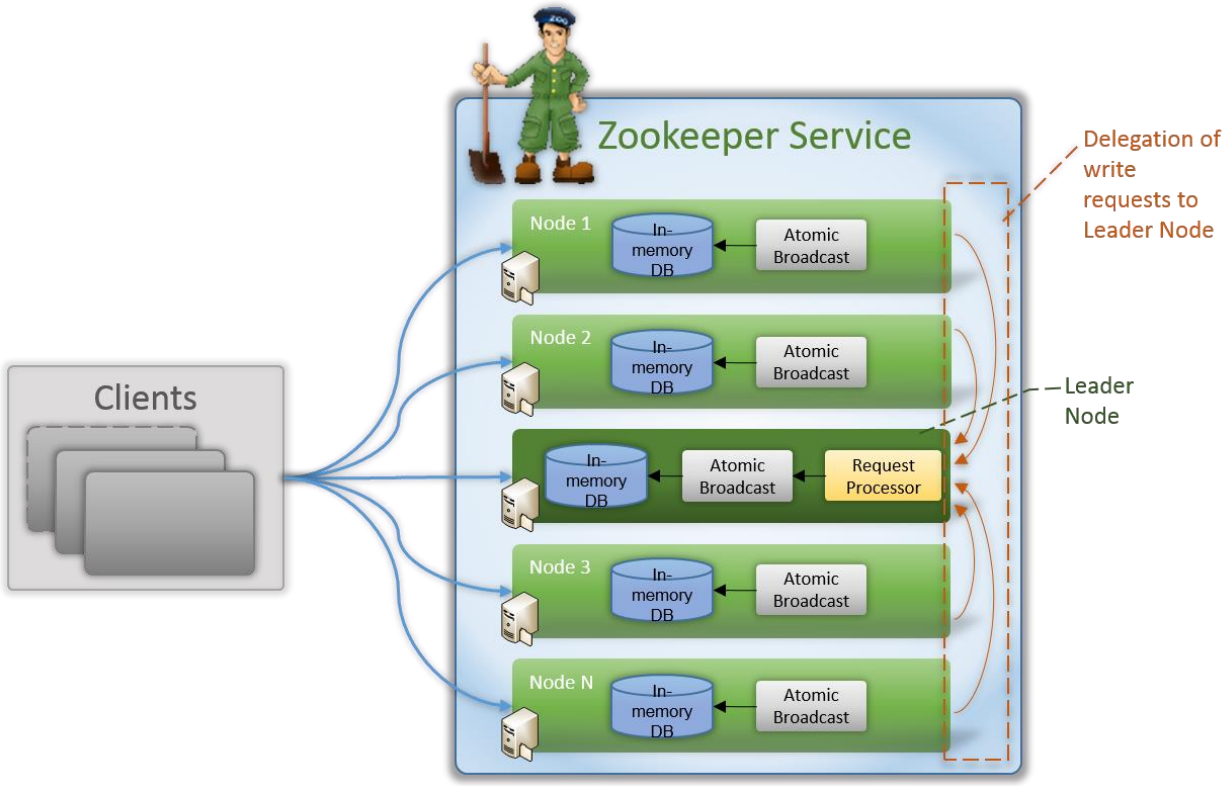
A ***distributed system*** is a system whose components are located on different networked computers, which communicate and coordinate their actions by passing messages to one another.

Apache ZooKeeper is a service used by a cluster (group of nodes) to coordinate between themselves and maintain shared data with robust synchronization techniques. ZooKeeper is itself a distributed application providing services for writing a distributed application.

The common services provided by ZooKeeper are as follows −

- Naming service − Identifying the nodes in a cluster by name. It is similar to DNS, but for nodes.
- Configuration management − Latest and up-to-date configuration information of the system for a joining node.
- Cluster management − Joining / leaving of a node in a cluster and node status at real time.
- Leader election − Electing a node as leader for coordination purpose.
- Locking and synchronization service − Locking the data while modifying it. This mechanism helps you in automatic fail recovery while connecting other distributed applications like Apache HBase.
- Highly reliable data registry − Availability of data even when one or a few nodes are down.



Client	Clients, one of the nodes in our distributed application cluster, access information from the server. For a particular time interval, every client sends a message to the server to let the sever know that the client is alive. Similarly, the server sends an acknowledgement when a client connects. If there is no response from the connected server, the client automatically redirects the message to another server.
Server	Server, one of the nodes in our ZooKeeper ensemble, provides all the services to clients. Gives acknowledgement to client to inform that the server is alive.
Ensemble	Group of ZooKeeper servers. The minimum number of nodes that is required to form an ensemble is 3.
Leader	Server node which performs automatic recovery if any of the connected node failed. Leaders are elected on service startup.

Follower	Server node which follows leader instruction.

What is Apache ZooKeeper Meant For?

Apache ZooKeeper is a service used by a cluster (group of nodes) to coordinate between themselves and maintain shared data with robust synchronization techniques. ZooKeeper is itself a distributed application providing services for writing a distributed application.

Workflow:

- Once a ZooKeeper ensemble starts, it will wait for the clients to connect.
- Clients will connect to one of the nodes in the ZooKeeper ensemble.
- It may be a leader or a follower node. Once a client is connected, the node assigns a session ID to the particular client and sends an acknowledgement to the client.
- If the client does not get an acknowledgment, it simply tries to connect another node in the ZooKeeper ensemble.
- Once connected to a node, the client will send heartbeats to the node in a regular interval to make sure that the connection is not lost.

The common services provided by ZooKeeper are as follows −

- Naming service − Identifying the nodes in a cluster by name. It is similar to DNS, but for nodes.
- Configuration management − Latest and up-to-date configuration information of the system for a joining node.
- Cluster management − Joining / leaving of a node in a cluster and node status at real time.
- Leader election − Electing a node as leader for coordination purpose.
- Locking and synchronization service − Locking the data while modifying it. This mechanism helps you in automatic fail recovery while connecting other distributed applications like Apache HBase.

- **Highly reliable data registry** – Availability of data even when one or a few nodes are down.

Distributed applications offer a lot of benefits, but they throw a few complex and hard-to-crack challenges as well. ZooKeeper framework provides a complete mechanism to overcome all the challenges. Race condition and deadlock are handled using **fail-safe synchronization approach**. Another main drawback is inconsistency of data, which ZooKeeper resolves with **atomicity**.

Benefits of ZooKeeper

Here are the benefits of using ZooKeeper –

- **Simple distributed coordination process**
- **Synchronization** – Mutual exclusion and co-operation between server processes. This process helps in Apache HBase for configuration management.
- **Ordered Messages**
- **Serialization** – Encode the data according to specific rules. Ensure your application runs consistently. This approach can be used in MapReduce to coordinate queue to execute running threads.
- **Reliability**
- **Atomicity** – Data transfer either succeed or fail completely, but no transaction is partial.

ZooKeeper node is referred as **znode**.

There are following two types of nodes in Zookeeper-

- Leader Node - Leader Node is the only node responsible for processing the write requests. All other nodes called followers simply delegate the client write calls to Leader node.**(We currently have two leader election algorithms in ZooKeeper: LeaderElection and FastLeaderElection)**

- We don't mark any node as leader while setting up Apache ZooKeeper cluster. It instead is elected internally among all the nodes of cluster. Apache ZooKeeper uses the concept of majority for same i.e. Node that gets highest number of votes is elected as Leader.

- This serves as the basis of recommendation that suggests to have odd number of nodes in a cluster for best failover and availability. E.g. if we create the cluster of four nodes and two nodes go offline for some reason. Apache ZooKeeper will be down as half of the nodes have gone offline as it is not possible to gain majority for Leader node election. However if we create the cluster of five nodes, even if two nodes go offline, Apache ZooKeeper will still be functional as we still have majority of nodes in service.

- Follower Nodes - All nodes other than Leader are called Follower Nodes. A follower node is capable of servicing read requests on its own. For write requests, it gets it done through Leader Node. Followers also play an important role of electing a new leader if existing leader node goes down.

Below are some of instances where Apache ZooKeeper is being utilized -

- Apache Storm, being a real time stateless processing/computing framework, manages its state in ZooKeeper Service

- Apache Kafka uses it for choosing leader node for the topic partitions

- Apache YARN relies on it for the automatic failover of resource manager (master node)

- Yahoo! utilties it as the coordination and failure recovery service for Yahoo! Message Broker, which is a highly scalable publish-subscribe system managing thousands of topics for replication and data delivery. It is used by the Fetching Service for Yahoo! crawler, where it also manages failure recovery.