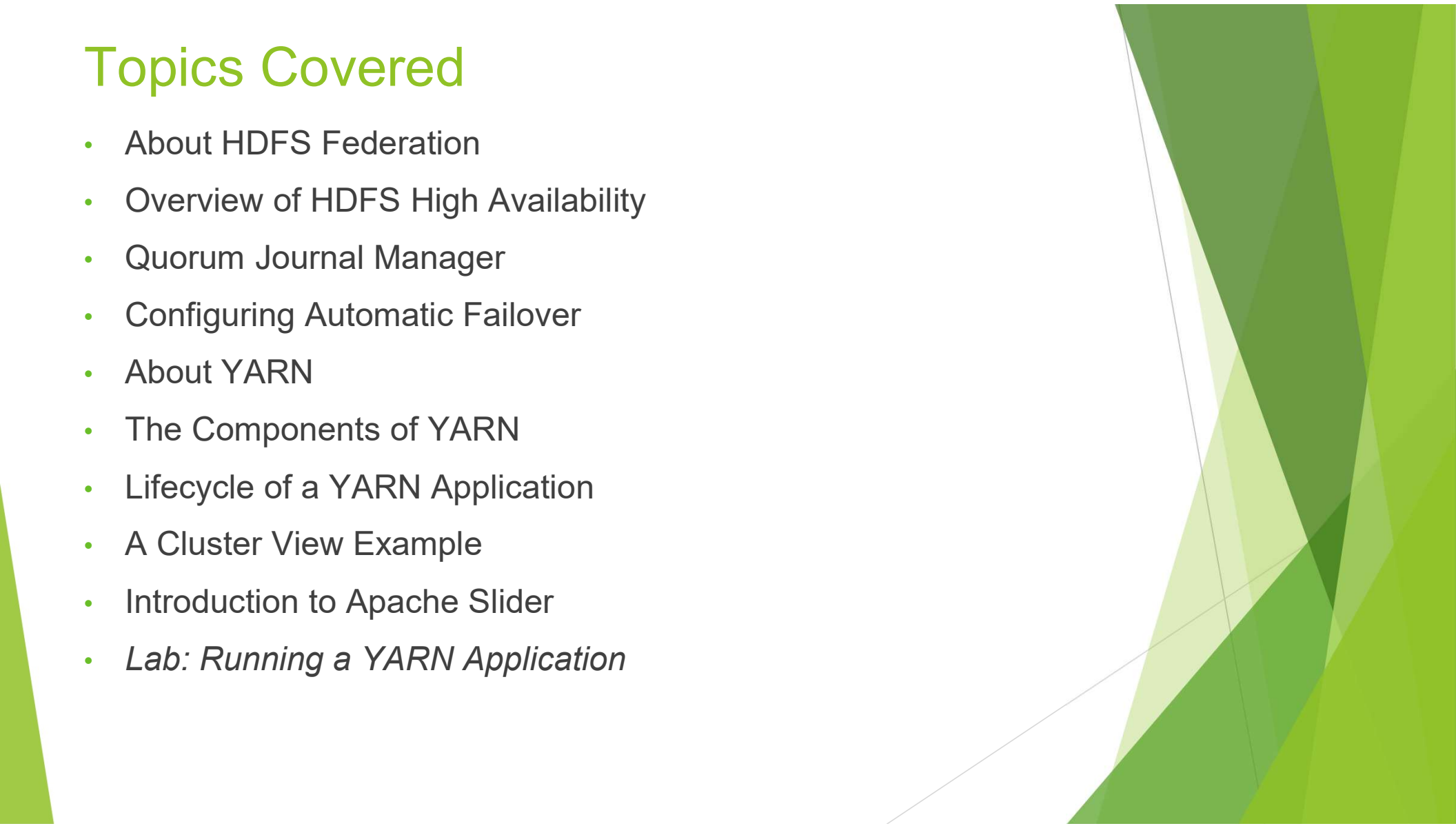
# Hadoop 2 and YARN

# Topics Covered

- About HDFS Federation
- Overview of HDFS High Availability
- Quorum Journal Manager
- Configuring Automatic Failover
- About YARN
- The Components of YARN
- Lifecycle of a YARN Application
- A Cluster View Example
- Introduction to Apache Slider
- *Lab: Running a YARN Application*

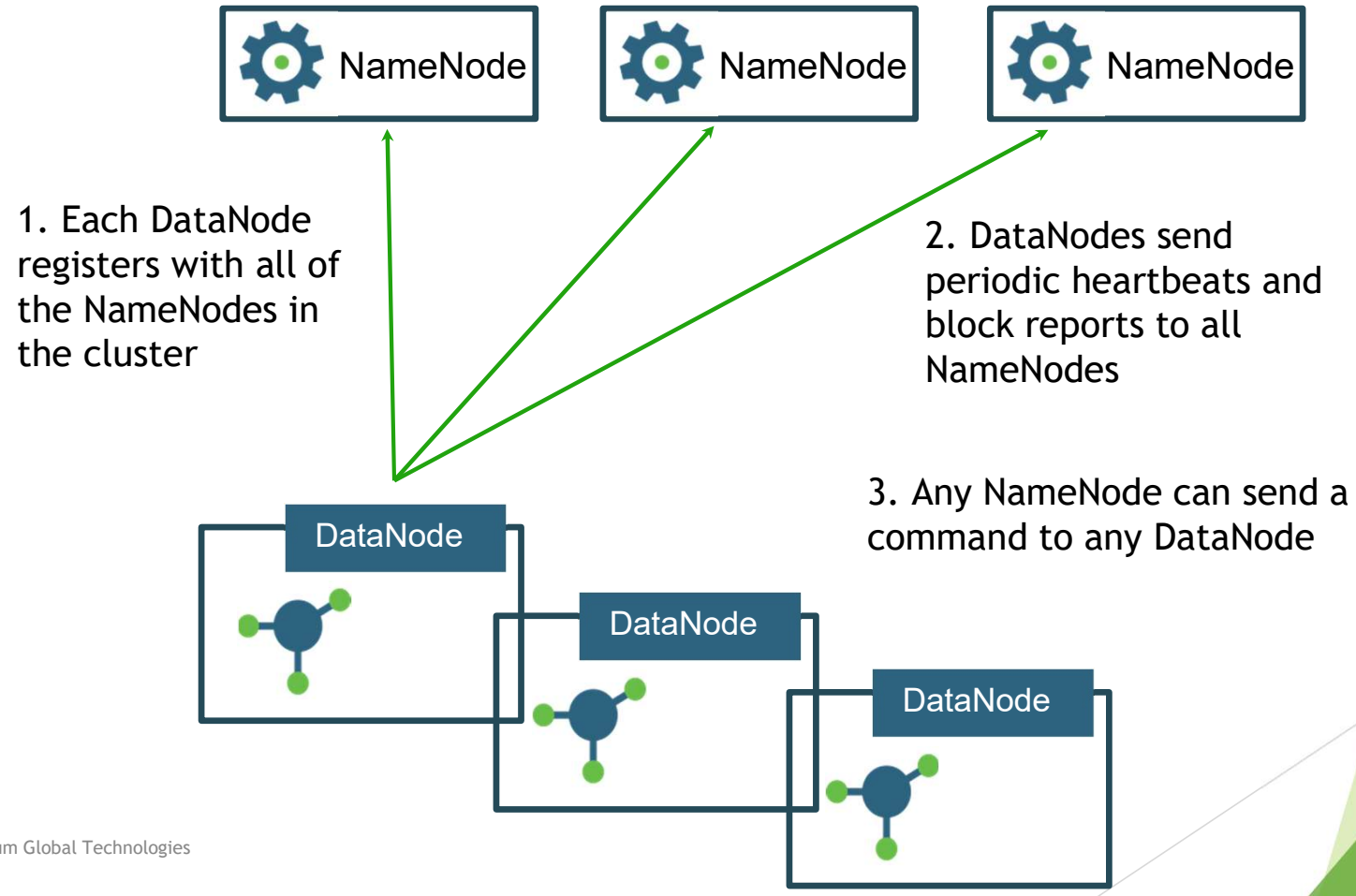


# About HDFS Federation

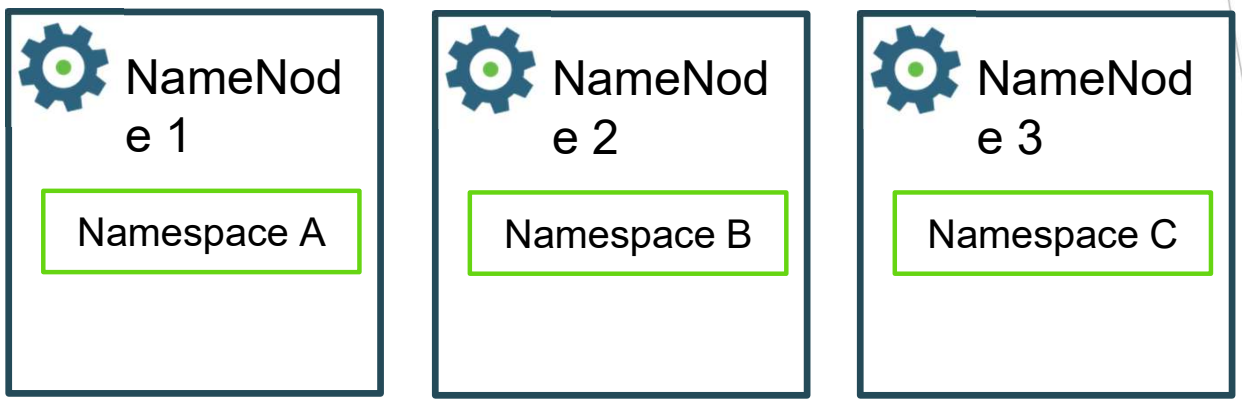
- According to Merriam-Webster's dictionary: a ***federation*** is an organization or group within which smaller divisions have some degree of internal autonomy

- **HDFS Federation** refers to the ability of NameNodes to work independently of each other

# Multiple Federated NameNodes



# Multiple Namespaces



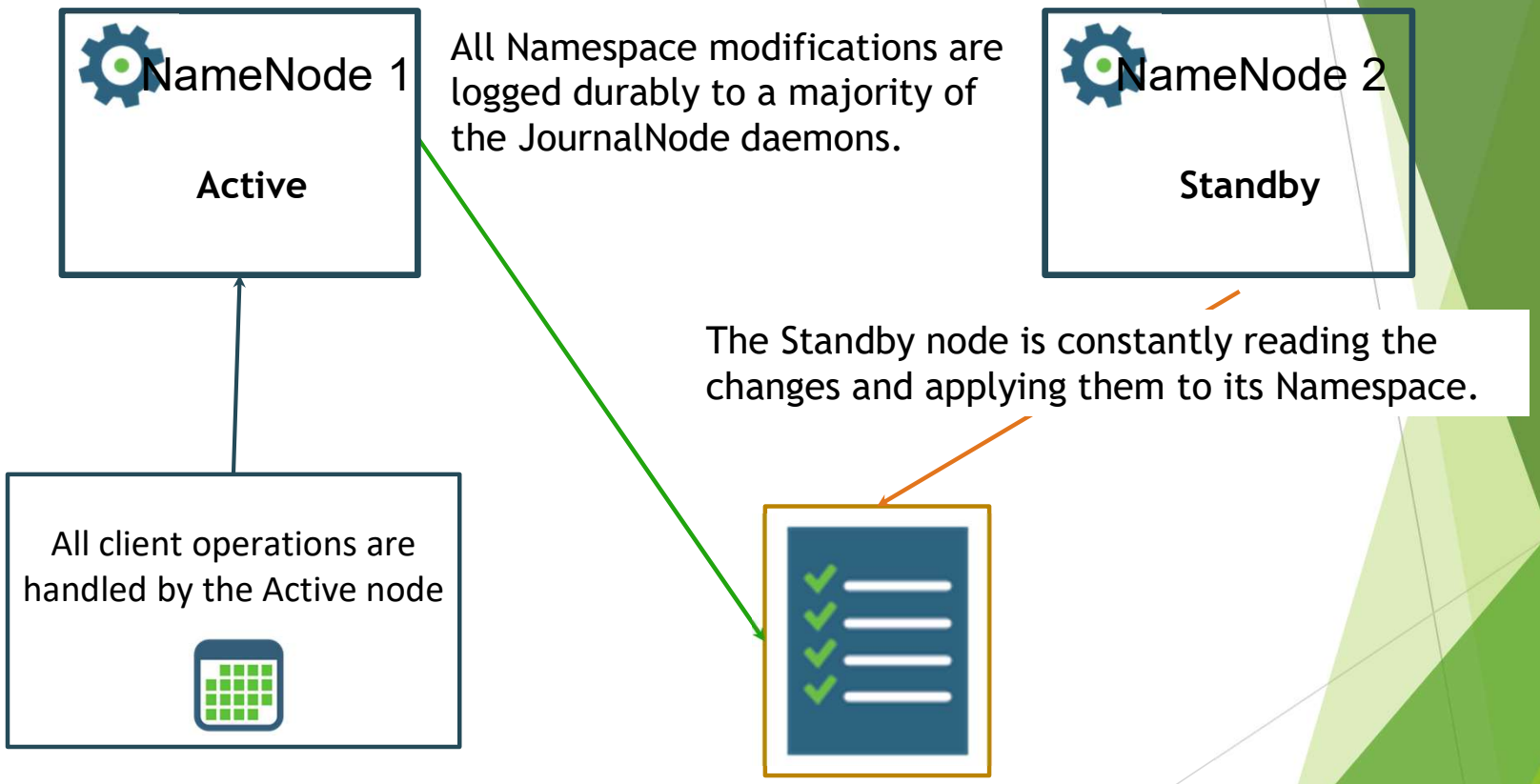

- Files and directories belong to a Namespace
- Prior versions of Hadoop only had a single Namespace
- Hadoop 2.x allows for multiple Namespaces
- A NameNode manages a single Namespace Volume

# Overview of HDFS HA

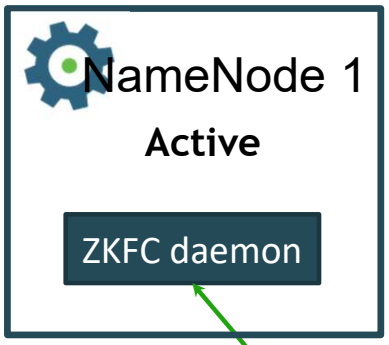
- In prior versions of Hadoop, the NameNode was a single point of failure that required additional tools to achieve HA

- In Hadoop 2.x, NameNode HA can be achieved using the built-in Quorum Journal Manager framework, Zookeeper, and the new Zookeeper Failover Controller processes

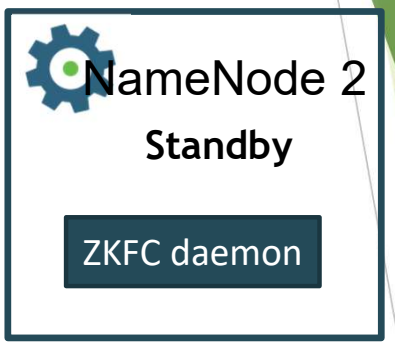
# Quorum Journal Manager



# Configuring Automatic Failover




ZKFC holds a lock to the Active NameNode. If that NameNode fails, the lock is made available.




The ZooKeeper daemons determine if a NameNode has failed. It also provides the lock that the ZKFC uses.

ZooKeeper instances.

# About YARN

YARN = Yet Another Resource Negotiator

YARN splits up the functionality of the JobTracker in Hadoop 1.x into two separate processes:

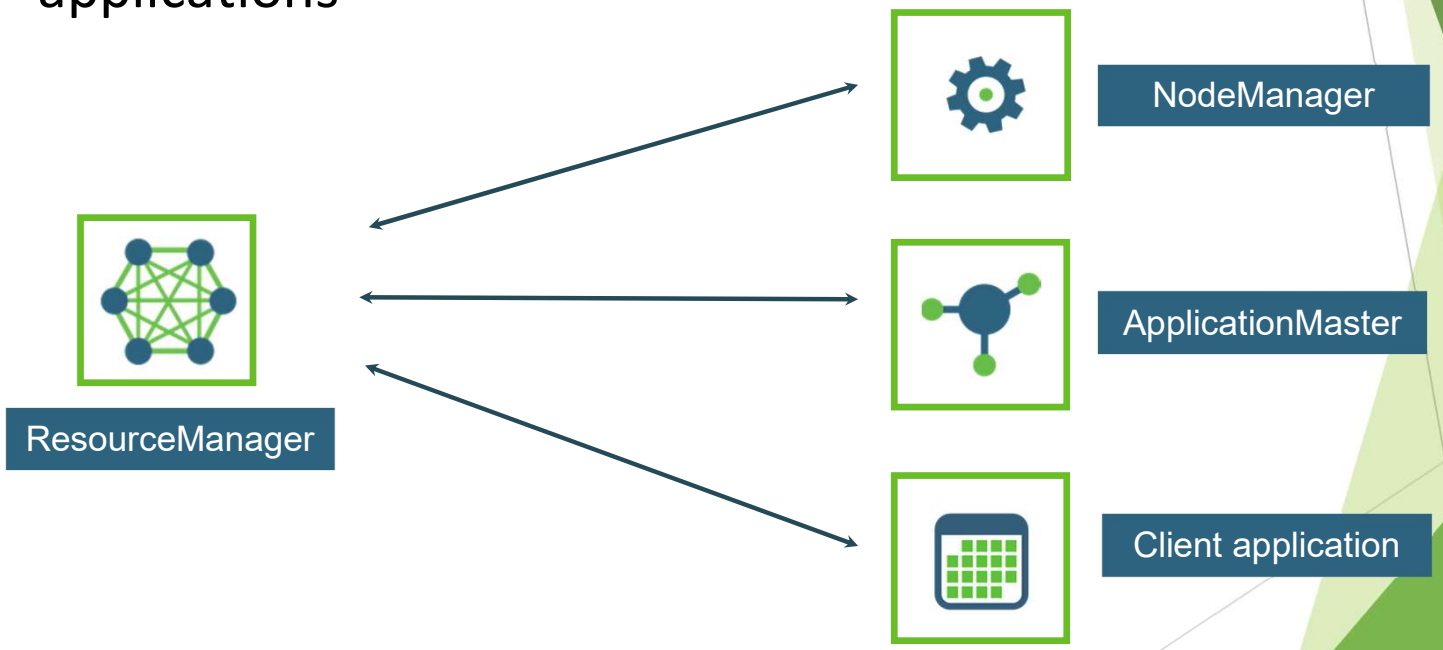
- **ResourceManager**: for allocating resources and scheduling applications
- **ApplicationMaster**: for executing applications and providing failover

# Open-source YARN Use Cases

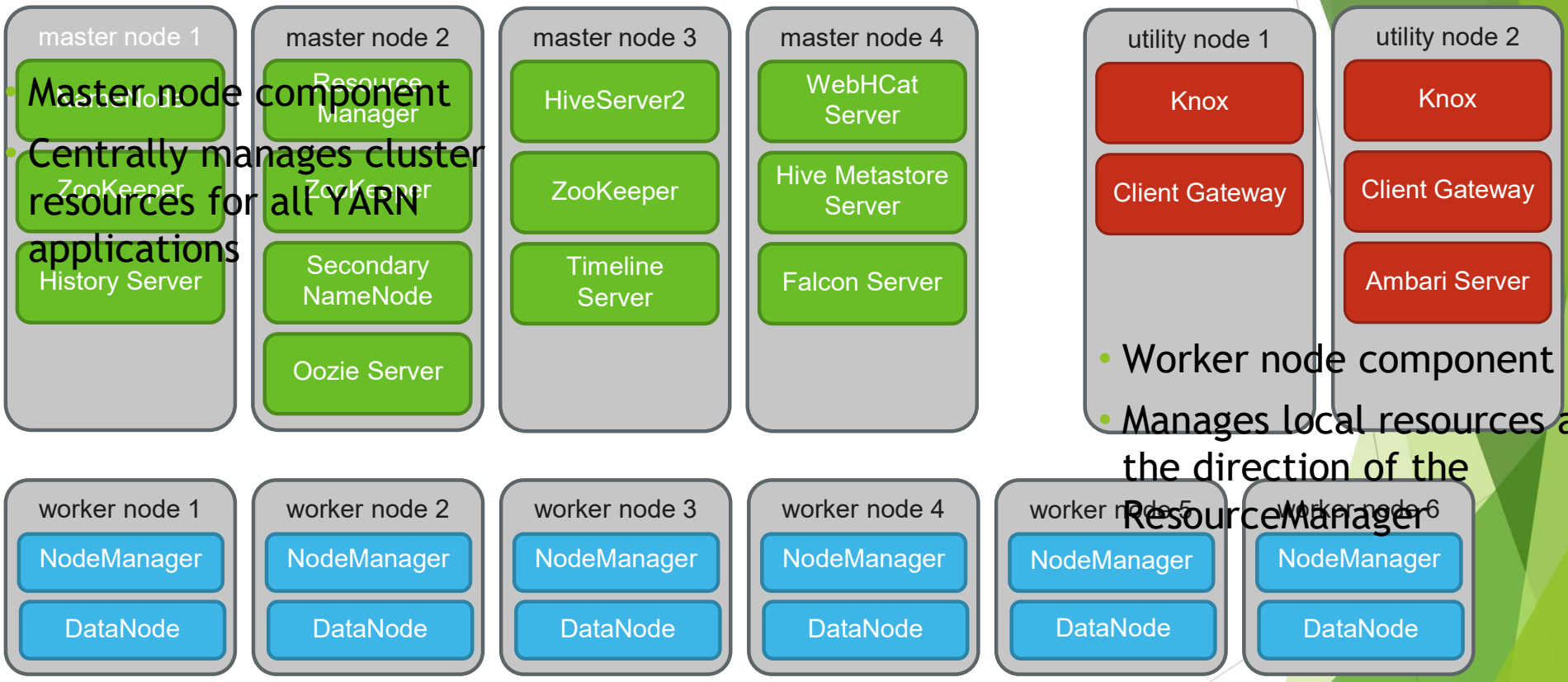
- **Tez:** improves the execution of MapReduce jobs
- **Slider:** deploy existing frameworks on YARN
- **Storm**: for real-time computing
- **Spark**: a MapReduce-like cluster computing framework designed for low-latency iterative jobs and interactive use from an interpreter
- **Apache Giraph**: a graph-processing platform

# The Components of YARN

The **ResourceManager** communicates with the **NodeManagers**, **ApplicationMasters**, and **Client** applications



# YARN Architecture – Big Picture View




- Master node component
- Centrally manages cluster resources for all YARN applications

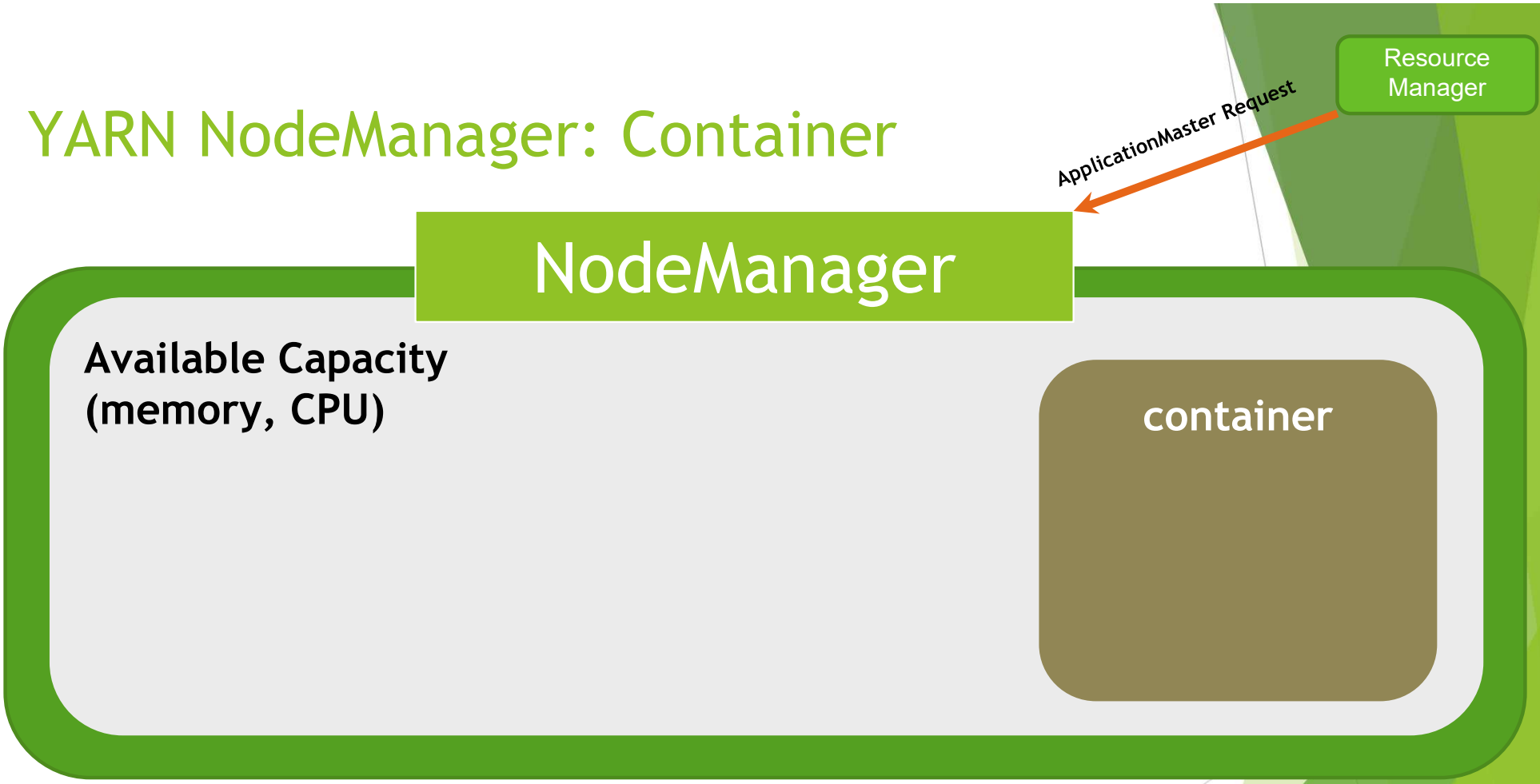
- Worker node component
- Manages local resources at the direction of the ResourceManager

# YARN NodeManager

## NodeManager

- Manages local CPU and RAM resources on behalf of requesting services
- Tracks node health and communicates status to the ResourceManager

# YARN NodeManager: Container

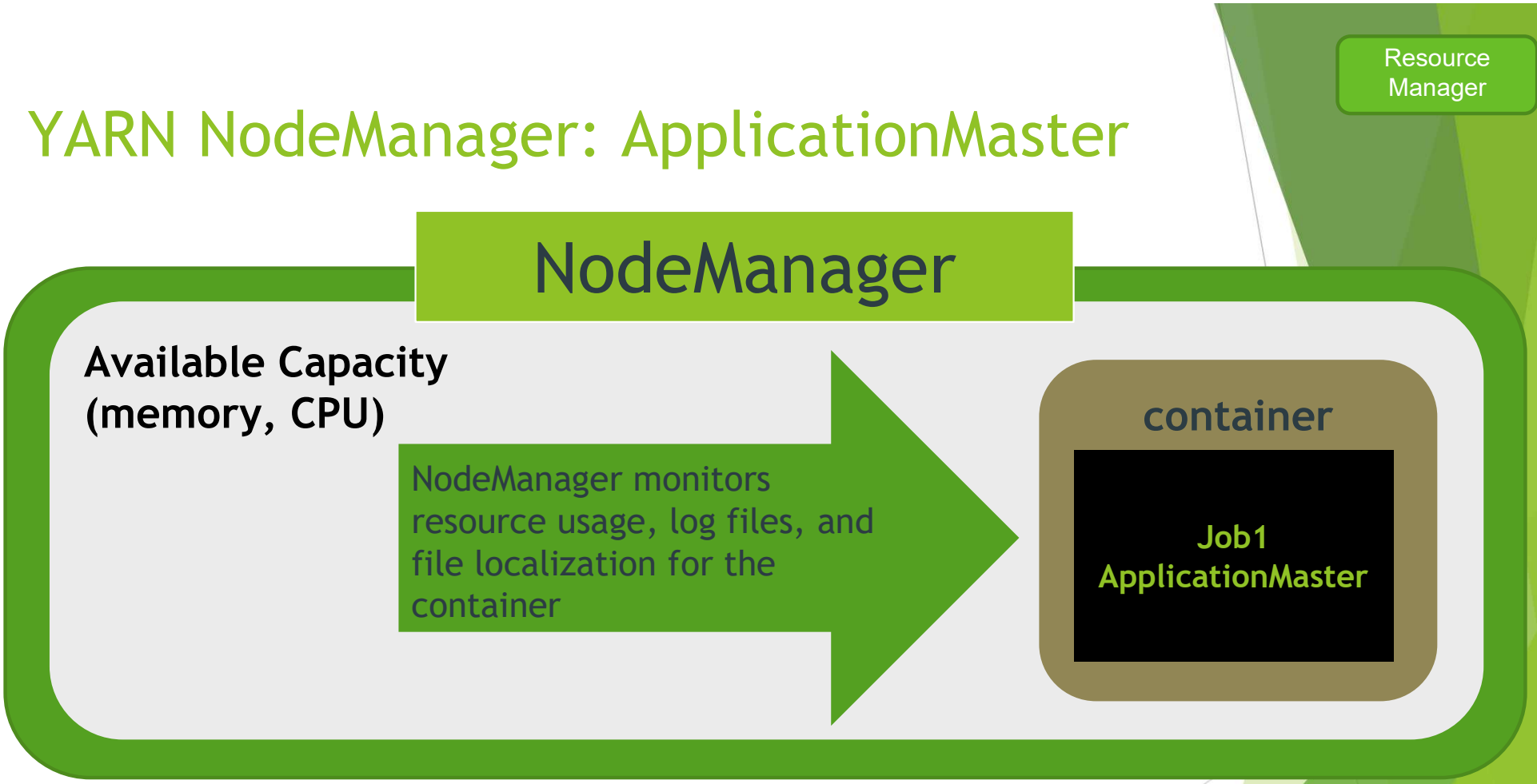


# Containers Defined

**container**

- Allocated RAM and CPU cores by the NodeManager
- Runs ApplicationMaster job
- New container spawned for each discrete job task

# YARN NodeManager: ApplicationMaster

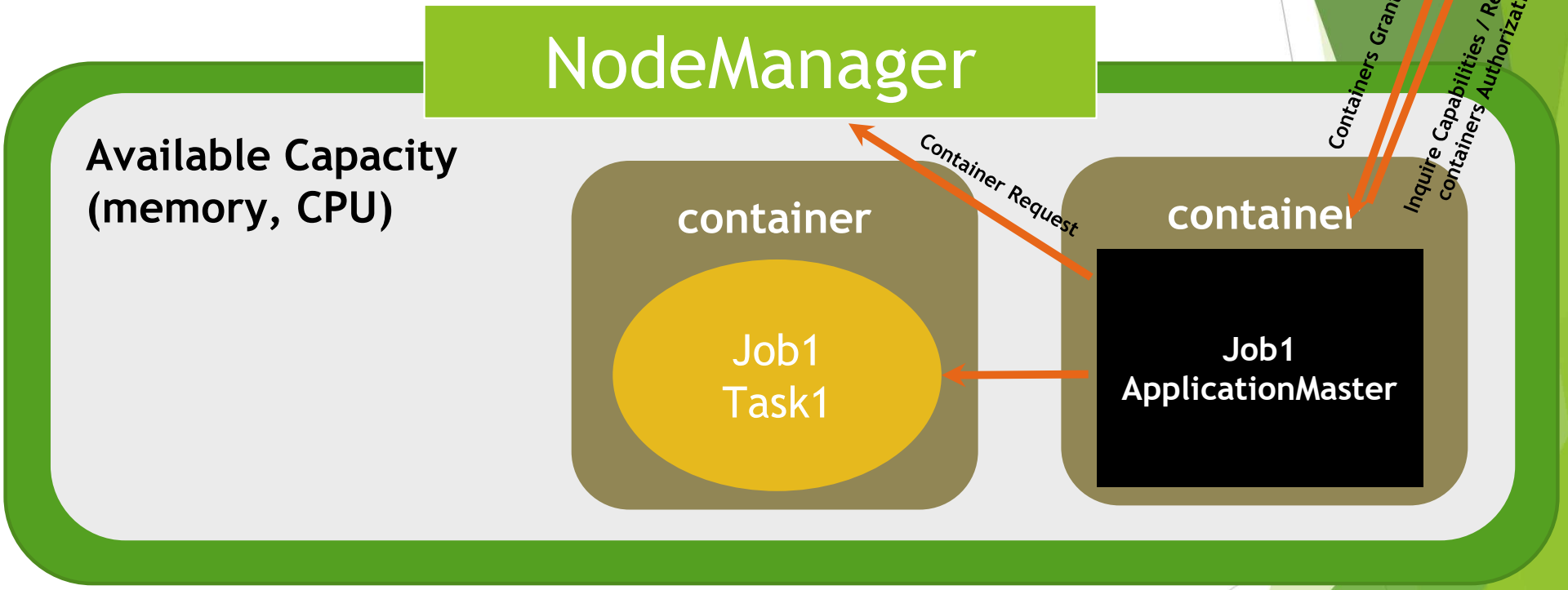


# ApplicationMasters Defined

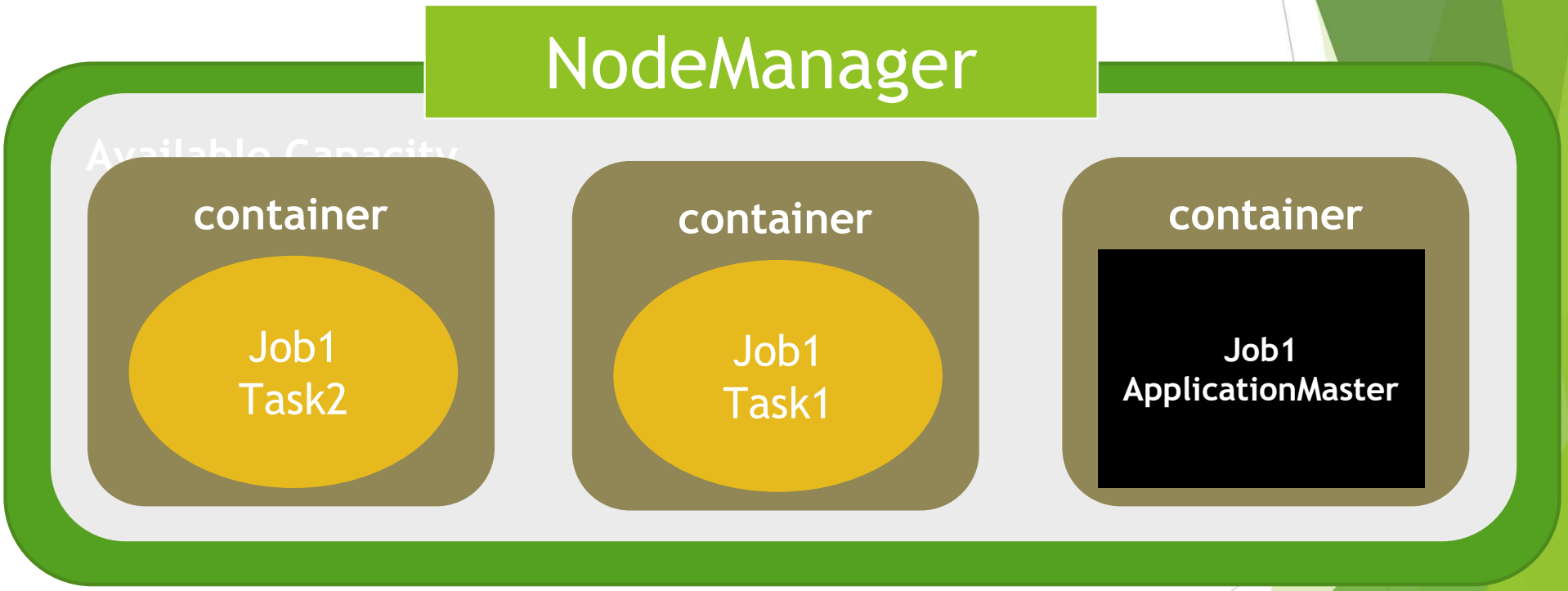
## ApplicationMaster

- Bootstrap process for YARN applications
- Negotiates for resources with ResourceManager
- Works with NodeManagers to configure and execute containers and monitors application resource consumption
- Provides application fault tolerance and thus significant horizontal scale capabilities

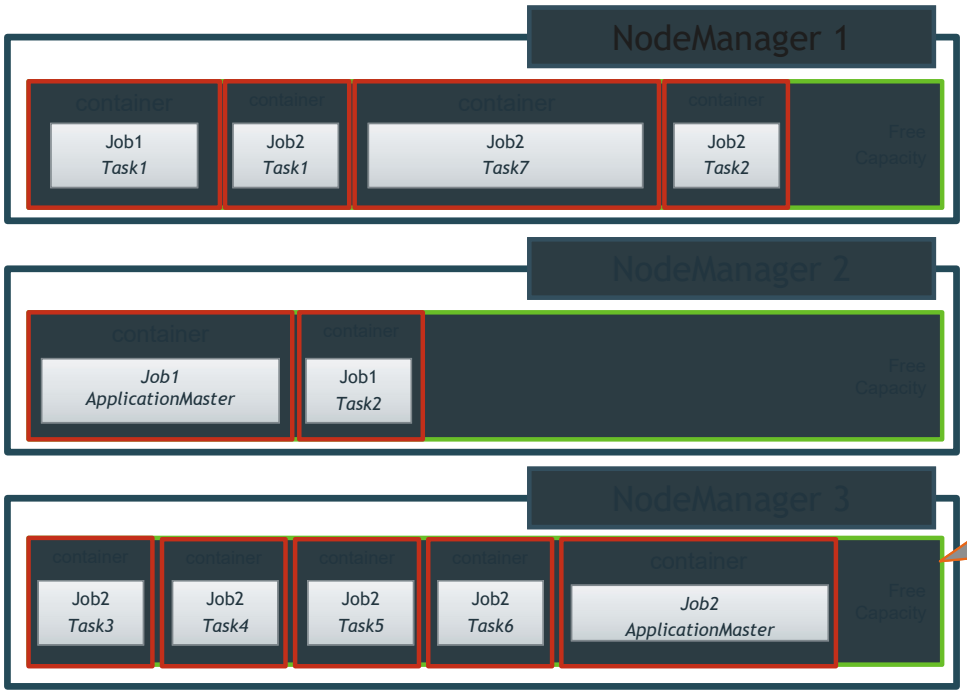
# YARN NodeManager: Job Scheduling



# YARN NodeManager: Additional Tasks



# Example Multi-node Resource Allocation Scenario




Default behavior is to move processing to data rather than copy data to nodes with available processing capacity

# YARN ResourceManager (Master Node)

## ResourceManager

**Scheduler**

- Controls global cluster resource usage
- Configurable by the Hadoop Administrator
- Enables multitenancy and Service Level Agreements

**Node Management**

- Monitor NodeManager state
- Submit ApplicationMaster Requests
- Verify container launch
- Monitor ApplicationMaster state

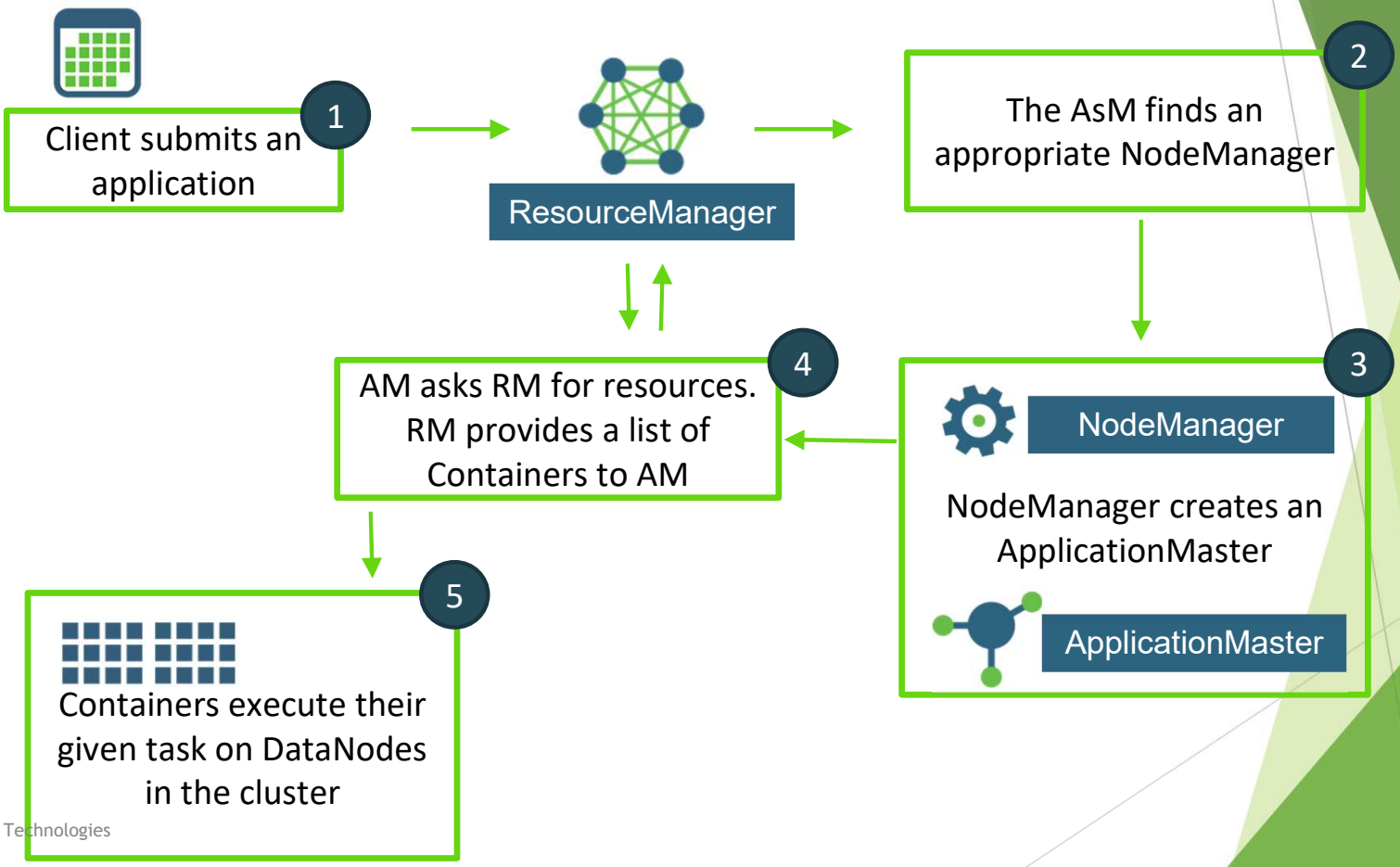
**Security**

- Web Application Proxy
- User ACLs
- Manages tokens to ensure validity of all container requests made by ApplicationMasters

# YARN Component Summary

| ResourceManager | NodeManager | Container | ApplicationMaster |
|---|---|---|---|
| Schedule global resources | Manage local memory and CPU allocation | Allocated RAM and CPU cores by NodeManager | YARN application bootstrap process |
| Enable multitenancy | | | Negotiate resources |
| Enable SLA enforcement | | | Provide application fault tolerance |
| Monitor and manage NodeManagers | Track and report on node health | | Work with NodeManager for container restart |
| Monitor and manage ApplicationMasters | Manage file localization for containers | Run ApplicationMasters and job tasks | |
| Monitor containers globally | Monitor and manage local containers | | Monitor job tasks and containers across cluster |
| Manage ACLs | | | |
| Manage Tokens | | | |

# Lifecycle of a YARN Application



# A Cluster View Example