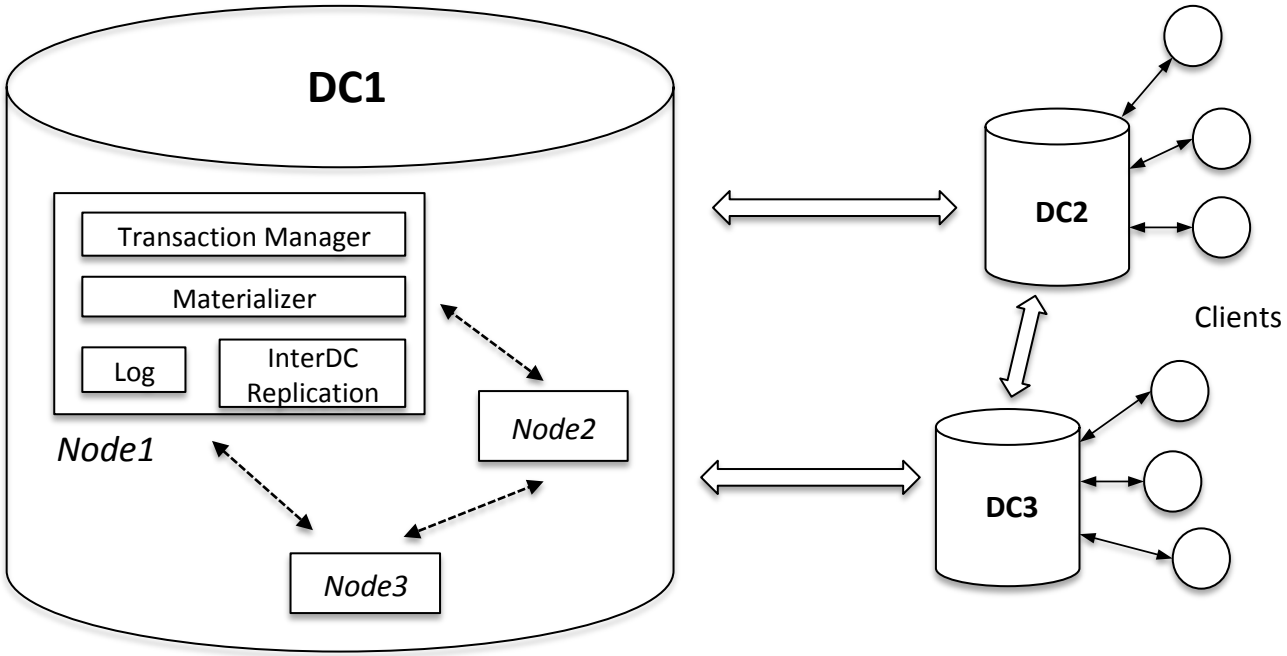
# Part III

## Antidote

# Antidote

- **AP** data store for geo-replication in the cloud
- Provides strongest form of consistency that is highly available, namely transactional causal consistency (Cure protocol)
- Supports programmer with comprehensive interface
  - Abstract data-types (CRDTs), including maps, sets, sequences, counters
  - Transactions operate on a consistent snapshot
  - Atomic update (e.g., allows non-normalised data)
- Use coordination only if its unavoidable (bounded counters)
  - Allows for Just-right-consistency

# Architecture




LIGHTKONE
Lightweight computation for networks at the edge

# Object API

```
let connection = connect(8087, "localhost")
```

*Establish connection*

```
connection.defaultBucket = "bucket1"
```

*Select bucket*

```
let s1 = connection.set("programmingLanguages")
await connection.update(s1.addAll(["Java","Erlang"]))
```

*Create new CRDT and perform update*

```
let res = await s1.read()
```

*Read current value*

LIGHTKONE

Lightweight computation for networks at the edge

# Transaction API

```
let set = connection.set("programmingLanguages")
{
        let tx = await connection.startTransaction()
        await tx.update(set.remove("Java"))
        await tx.update(set.add("Kotlin"))
        await tx.commit()
}{
        await connection.update([
        set.remove("Java"),
        set.add("Kotlin")])
}
```

*Variant 1: dynamic transaction*

*Variant 2: static transaction / batch updates*

LIGHTKONE

Lightweight computation for networks at the edge

# Conclusion: Part III

- Antidote provides Just-right consistency
  - Transactional Causal Consistency for AP-compatible invariants
  - Bounded Counters for CAP-sensitive invariants
- Supports programmer with rich interface
  - Transactions with snapshot reads and atomic updates
  - CRDTs avoid conflicting updates
- Documentation
  - http://antidotedb.org
- Code repository
  - https://github.com/SyncFree/antidote

LIGHTKONE
Lightweight computation for networks at the edge