BigTable

Min Cha
minslovey@gmail.com

What is BigTable?

- A distributed storage system based on GFS.
- Designed to scale to a very large size.
- Dynamic control over data layout and format.

Data Model

- 3D table -> row * column * timestamp
- Sparse, distributed, persistent multi-dimensional sorted map.

Data Model - Row

- The row like the row of RDB.
- Arbitrary strings in 64KB.

Atomic

Lexicographic order

Data Model – Row Atomic

- Every read or write of data under a single row key is atomic.
- So, clients should not worry as concurrent updates to the same row.

Data Model – Row Tablet

- BigTable maintains data in lexicographic order by row key. The row range for a table is dynamically partitioned. Each row range is called a tablet.
- 'maps.google.com/index.html' is stored as 'com.google.maps/index.html'
- When we read the row range for some domain, it might be more efficient due to lexicographic order as above example.

Data Model – Column Families

- The category of column keys.
- A BigTable could have numbers of column families.
- The syntax of a column key is as following. 'family:qualifier'
- A column family Itself could be a column key. ex. 'content:'

Data Model - Timestamps

- The multiple versions of same data.
- 64bit integer.
- It could be current microseconds or explicitly assigned from client applications.
- Old timestamps could be removed by garbage-collection.

API

- The Bigtable API provides functions for creating and deleting tables and column famlies.
- A BigTable could be used as an input source and an output target for MapReduce jobs.

Building Blocks

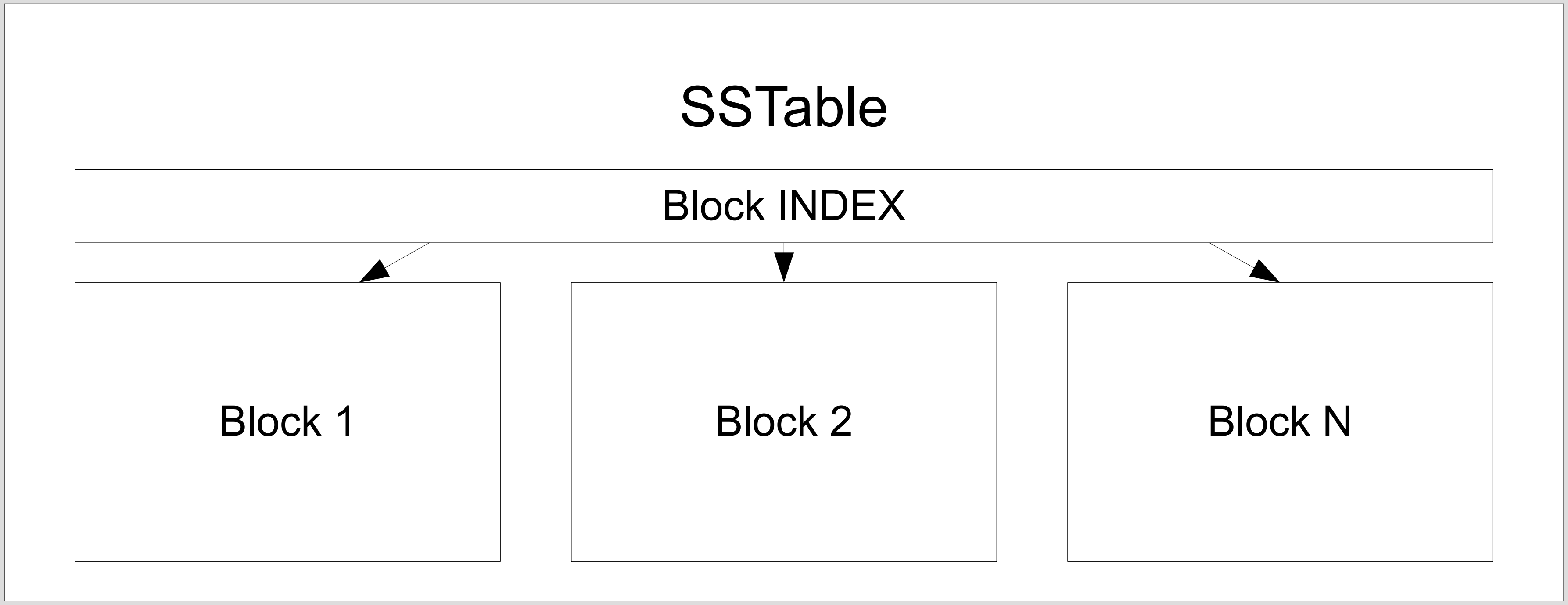
GFS

Chubby

SSTable

Building Blocks SSTable

- This is used internally to store Bigtable data.
- Look up using a key, iterate over all key/value pairs in specified key range.



Building Blocks
Chubby

- Highly-available and persistent distributed lock service.
- Chubby provices namespace that consists of directories and small files. They can be used as a lock.
- We could think of Chubby as the global lock repository of BigTable.

Implementation

- There are three major components.

Client Library

Master Server

Tablet server

Implementation
Master Server

- Assigning tablets to table servers.
- Detecting the addition and expiration of tablet servers.
- Balancing tablet-server load.
- Garbage collecting of files in GFS.

Implementation
Tablet Server

- Managing a set of tablets.
- Handling read and write requests to the tablets.
- Splitting tablets that have grown too large.

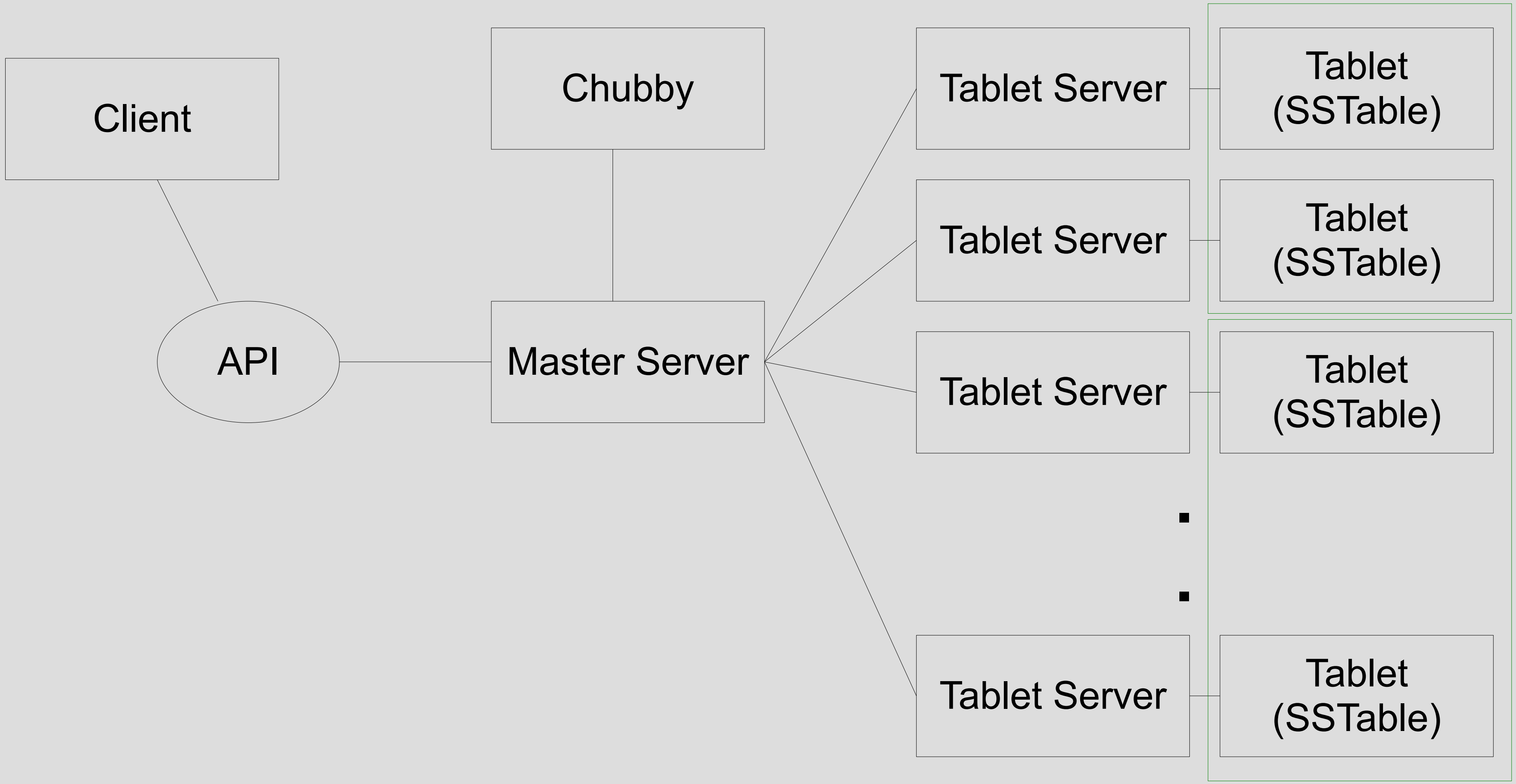
Implementation
Tablet Location

- Three-level hierarchy analogous to that of a B+ tree.
- Root tablet, METADATA tablets, UserTables

Implementation Memtable

- The recently committed logs are stored in memory in a sorted buffer called a memtable.
- When a read operation arrives, it is executed on a merged view of the sequence of SSTables and the memtable.
- When a write operation arrives, it is written to the commit log in memtable.
- When the memtable size reaches a threshold, it is converted to an SSTable and written to GFS. It`s called as Compactions.

Summary


Client
API
Chubby
Master Server
Tablet Server
Tablet Server
Tablet Server
Tablet Server
Tablet (SSTable)
Tablet (SSTable)
Tablet (SSTable)
Tablet (SSTable)