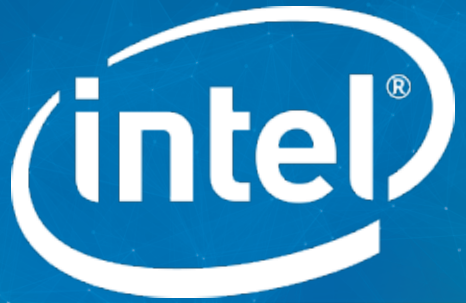
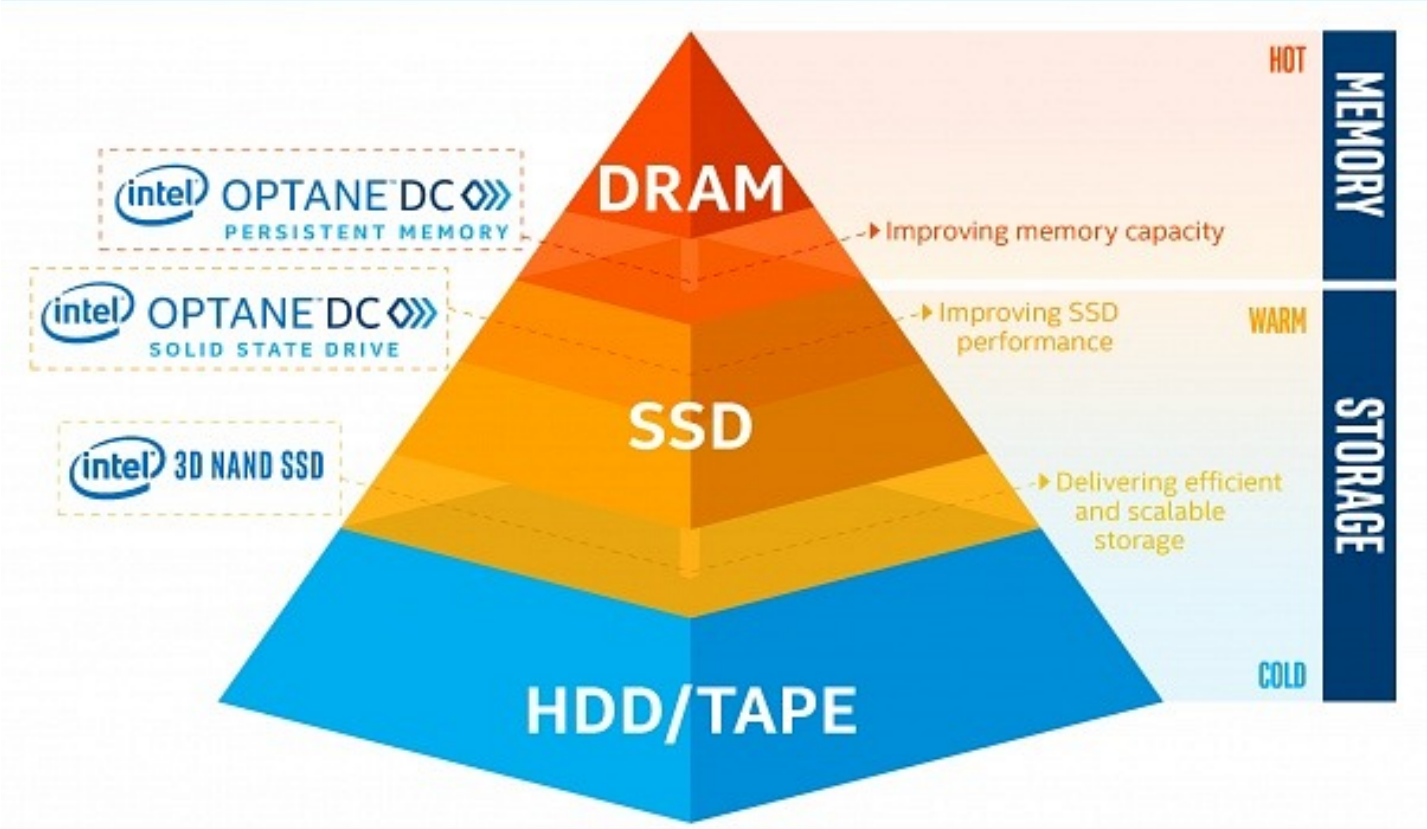
Persistent Memory Use Cases in Modern Software Architectures

Olasoji Denloye
SW Engineer
Intel Corporation



Persistent Memory

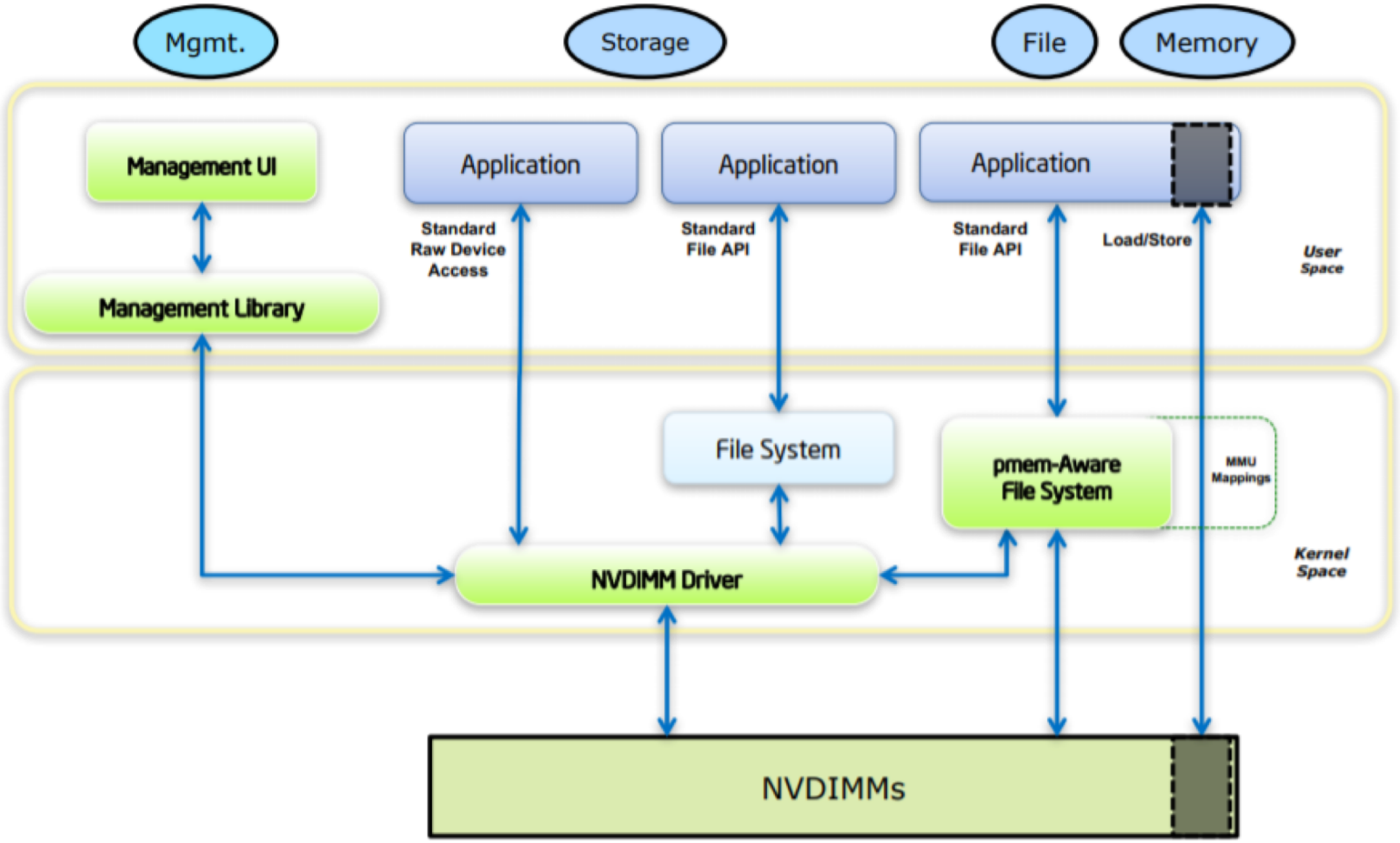
REIMAGINING THE DATA CENTER MEMORY AND STORAGE HIERARCHY



Properties of Persistent Memory

- Byte-addressable like DRAM
- Direct user-level access
- Lowers DRAM footprint
- Works with both File and Memory APIs
- Multiple Modes
  - Memory Mode - Persistent Memory as Main Memory
  - App Direct Mode - OS aware of Persistent Memory

The SNIA NVM Programming Model



Common Use Cases

- Caches
  - Data structures for fast lookup
- Stores
  - Device for persisting data
- Buffers
  - Temporary data storage

- Volatile Memory
- Persistent Memory

Caches

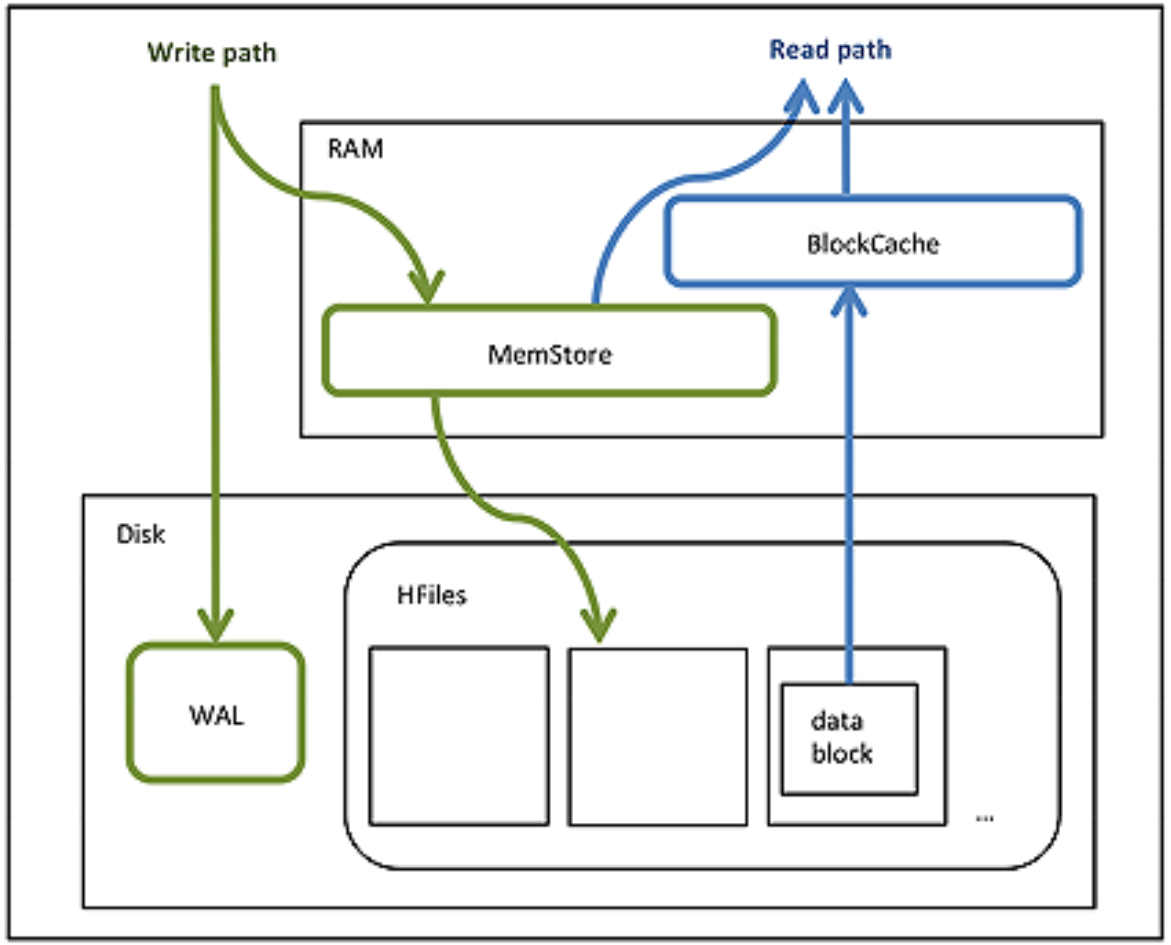
Caches

- Caches are fast and lightweight data structures
- Typically live on DRAM for speed
- Constrained by DRAM size
- Persistent Memory provides larger capacities than DRAM
- Faster restart times

HBase BucketCache

- Manages buckets of memory containing fixed size blocks
- Moved the cache from DRAM to Persistent Memory
- Uses mapped file based allocator

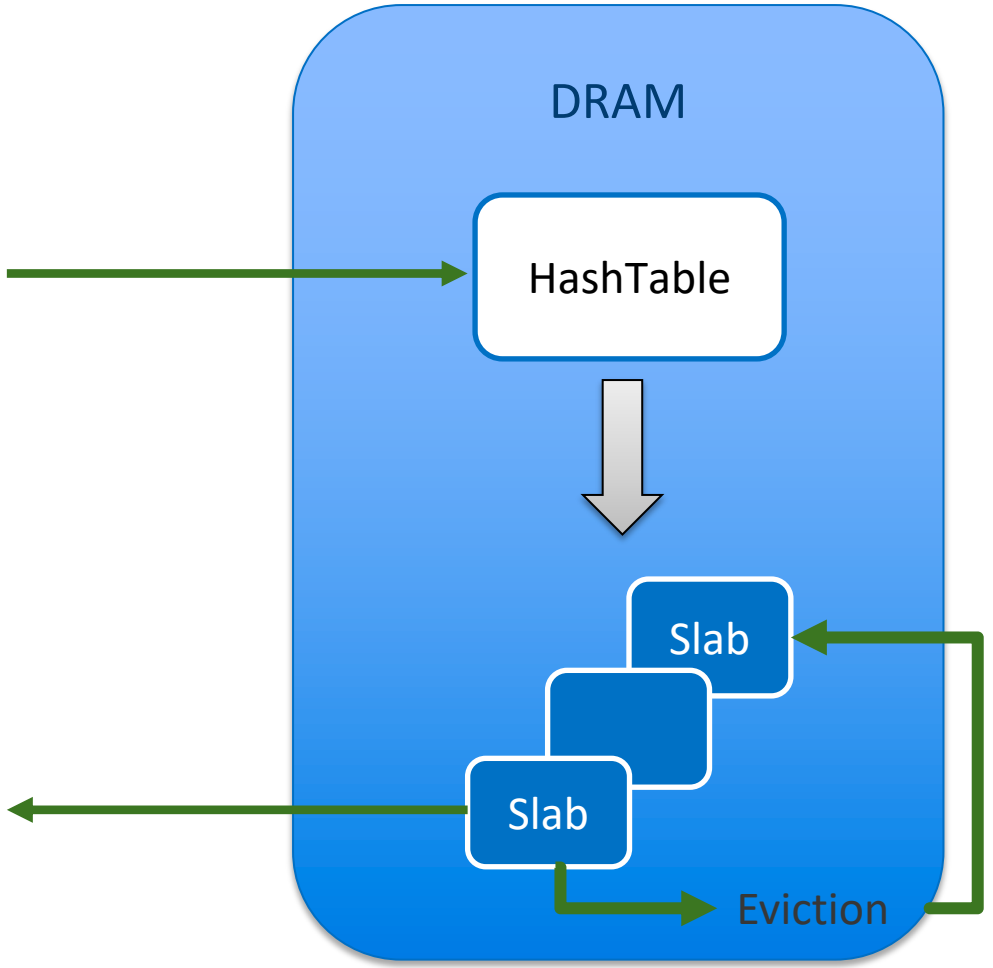
https://issues.apache.org/jira/browse/HBASE-21874



source: https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.1.4/hbase-data-access/content/overview-hbase-io.html

MemcacheD – DRAM

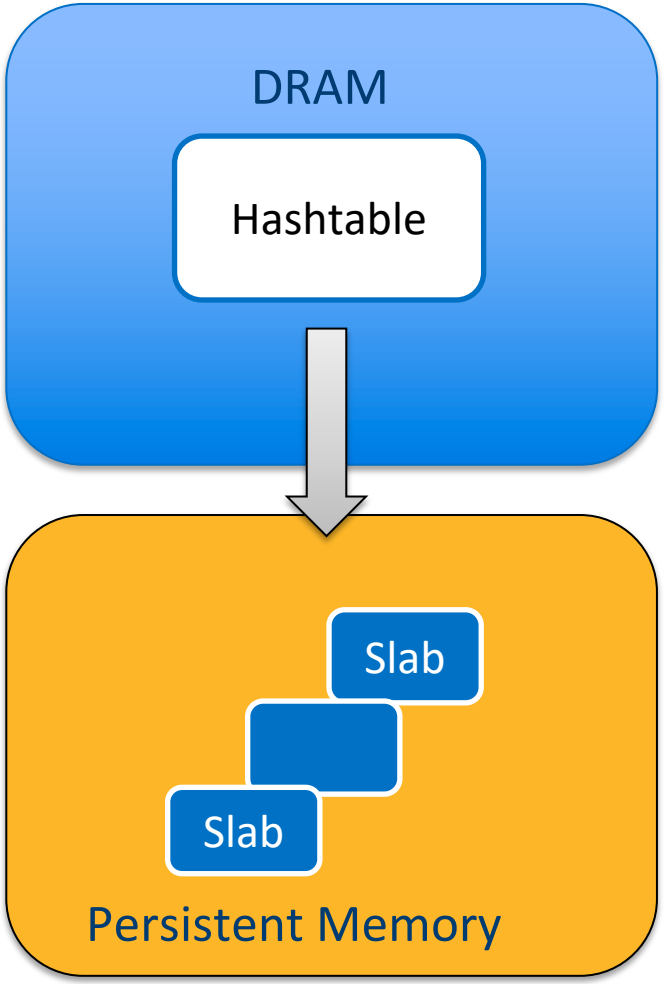
- In memory Key-Value store
- Used as a cache
- Designed to be simple and fast



MemcacheD – Restartable Cache

- Custom mapped file allocator
- Hybrid data structure
  - *Hashtable on DRAM*
  - *Slabs on Persistent Memory*
- Restartable
  - *Flush CPU caches on controlled shutdown*
  - *Rebuild Hashtable on restart*

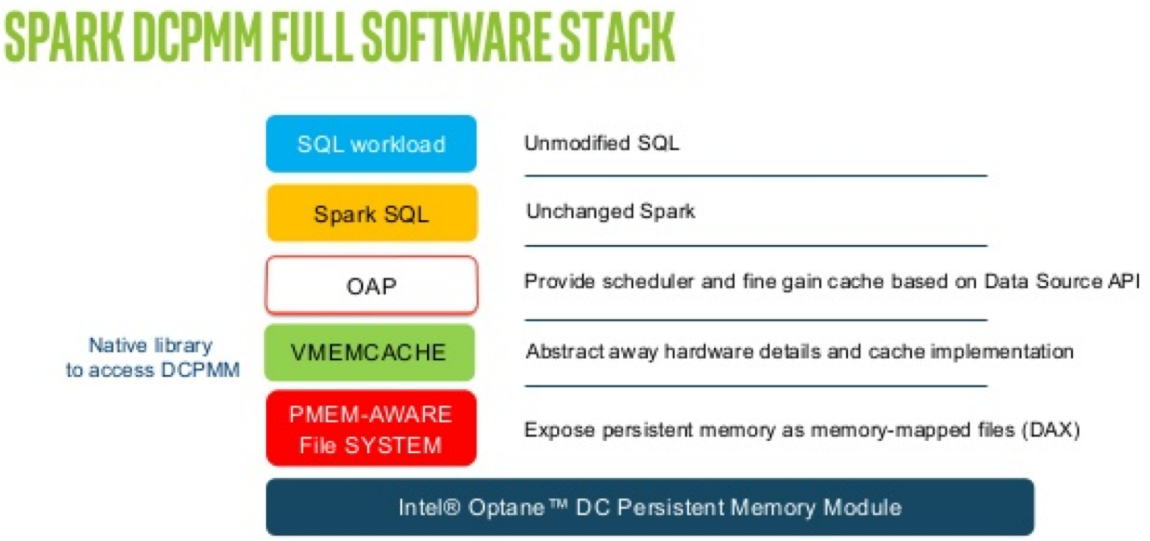
https://github.com/memcached/memcached/wiki/WarmRestart



Spark SQL

Spark OAP (Optimized Analytics Platform)

- OAP is a SparkSQL accelerator
- IO cache
- Uses Persistent Memory Development Kit (PMDK) : libvmemcache
  - Open source
  - Volatile LRU cache
  - Keys in DRAM, values on PMEM



https://github.com/Intel-bigdata/OAP

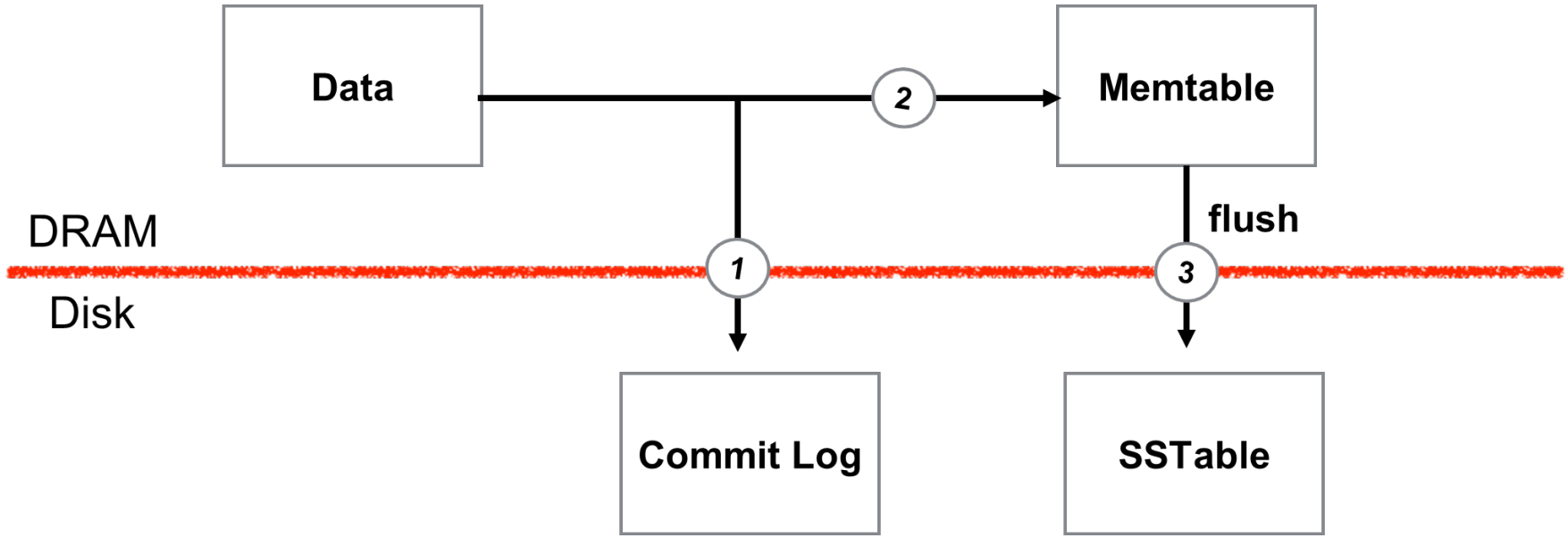
Storage

Storage

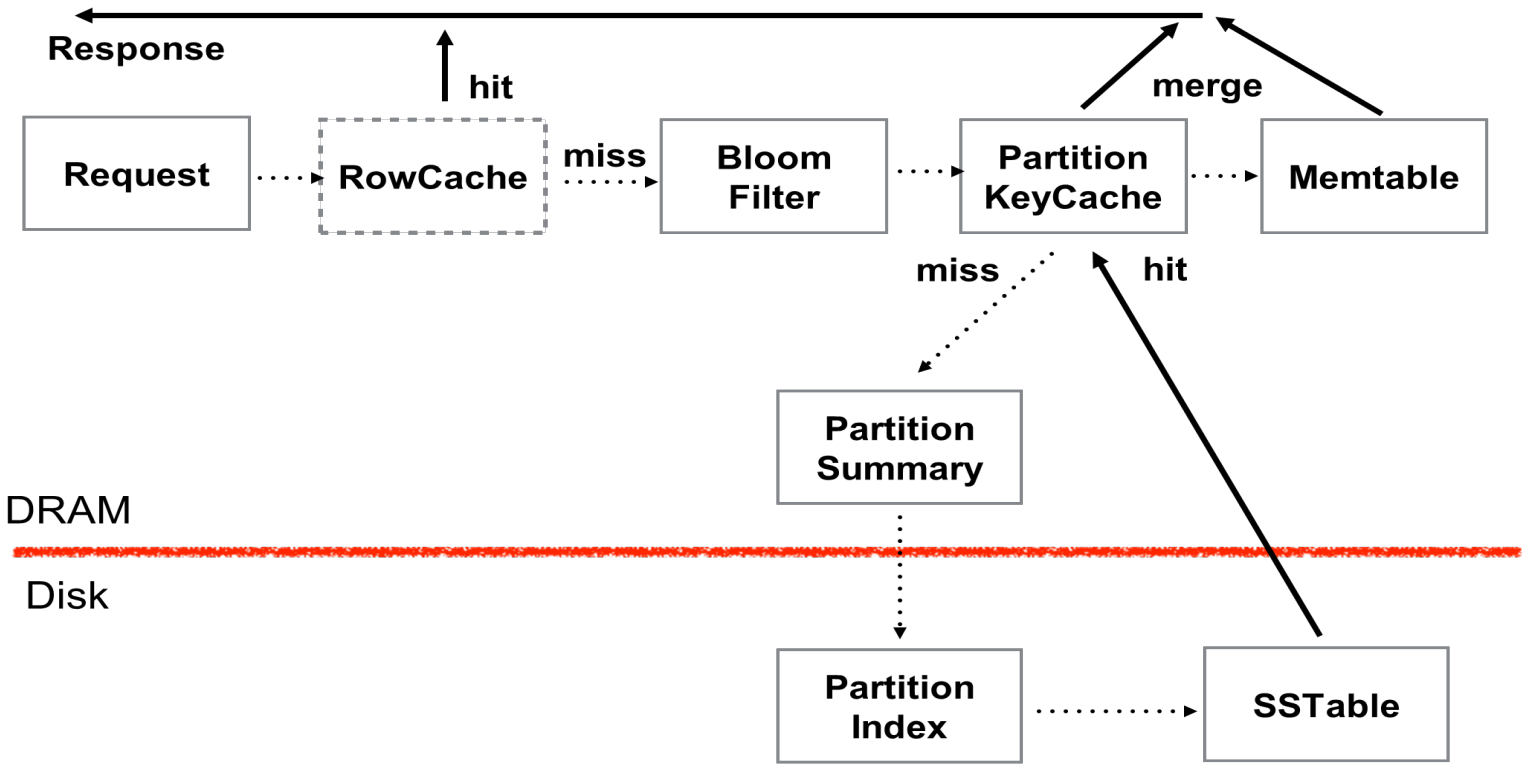
- Persists data
- Large capacity
- Typically SSDs or HDDs

- Persistent Memory is faster than NVME drives
- No need for serialization/deserialization
- Update in place
- Simpler code

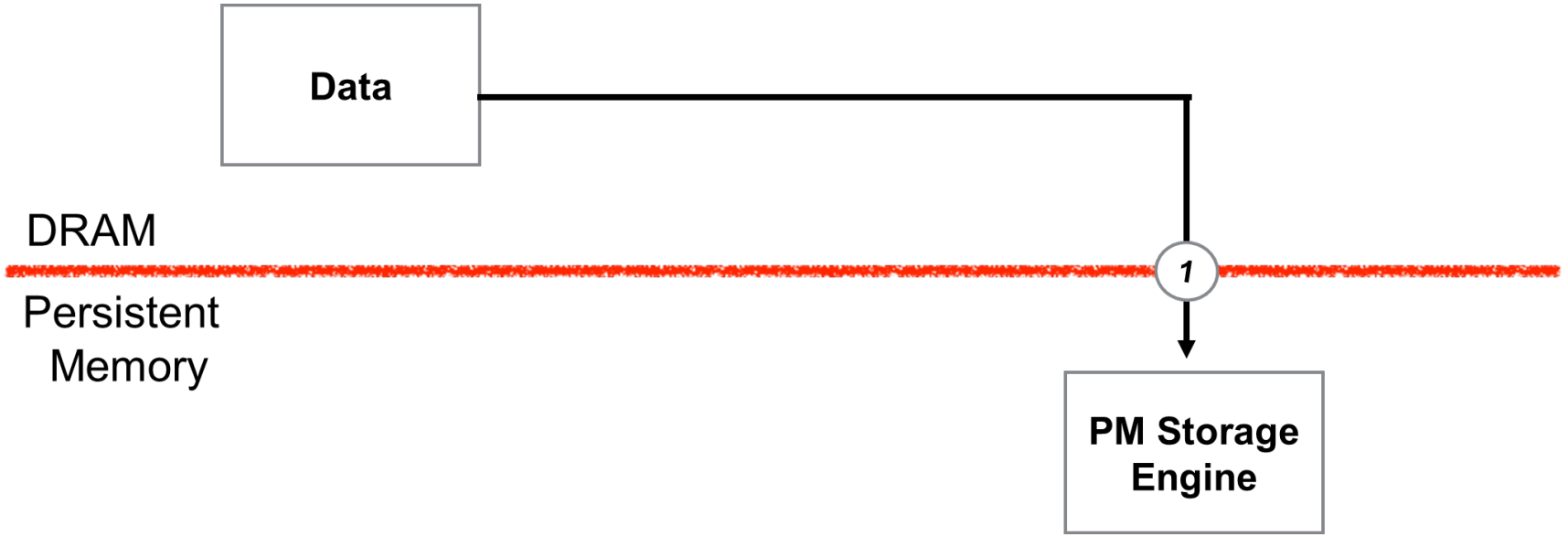
Cassandra Write Path



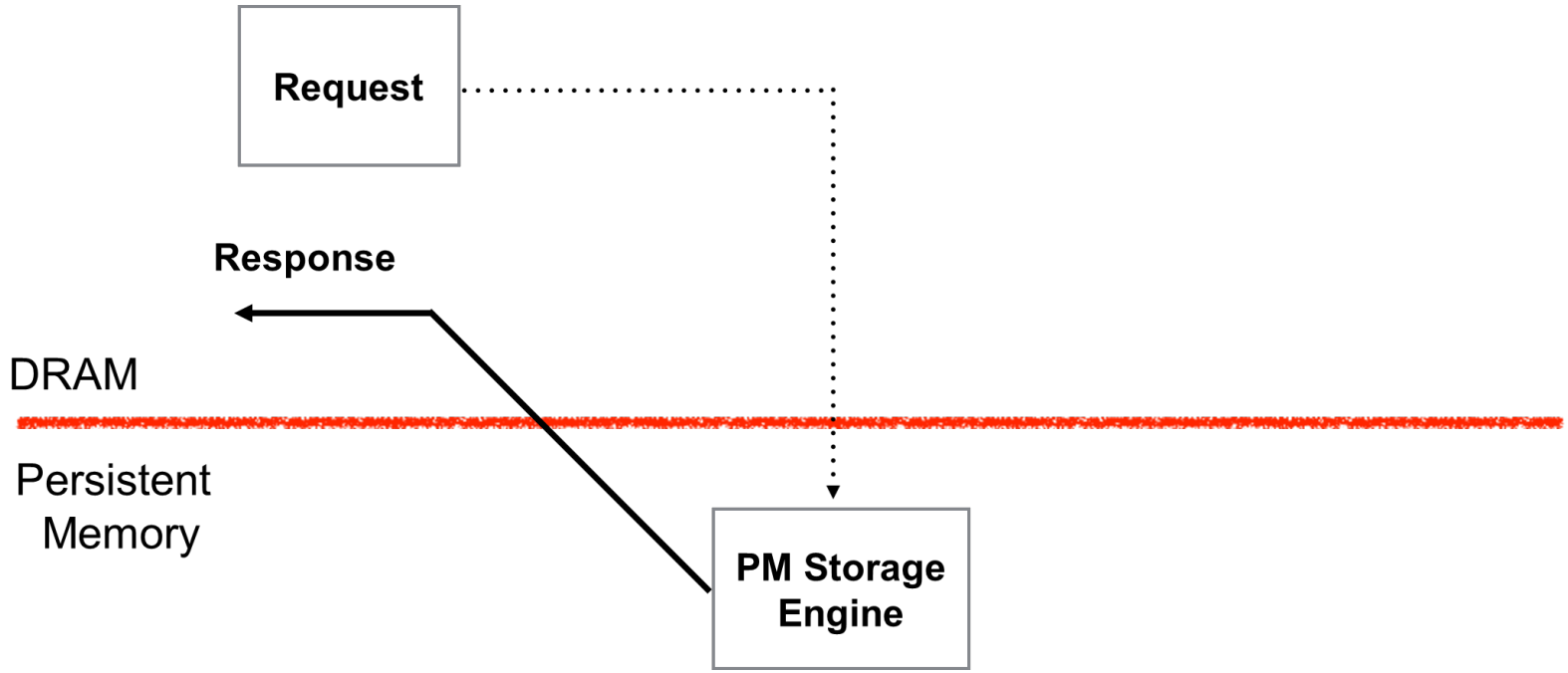
Cassandra Read Path



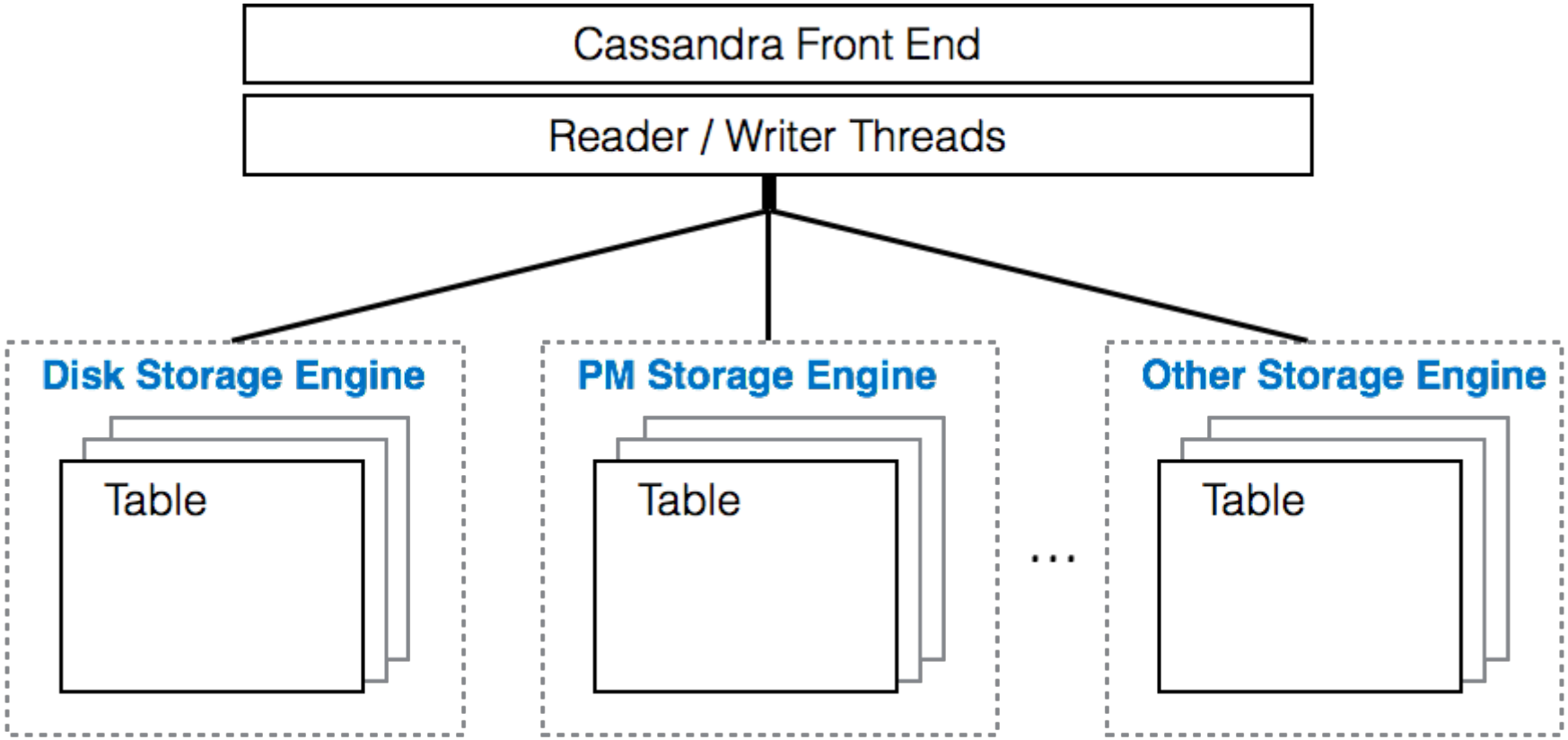
Write Path – Persistent Memory Storage



Read Path – Persistent Memory Storage



Cassandra Pluggable Storage Engine API

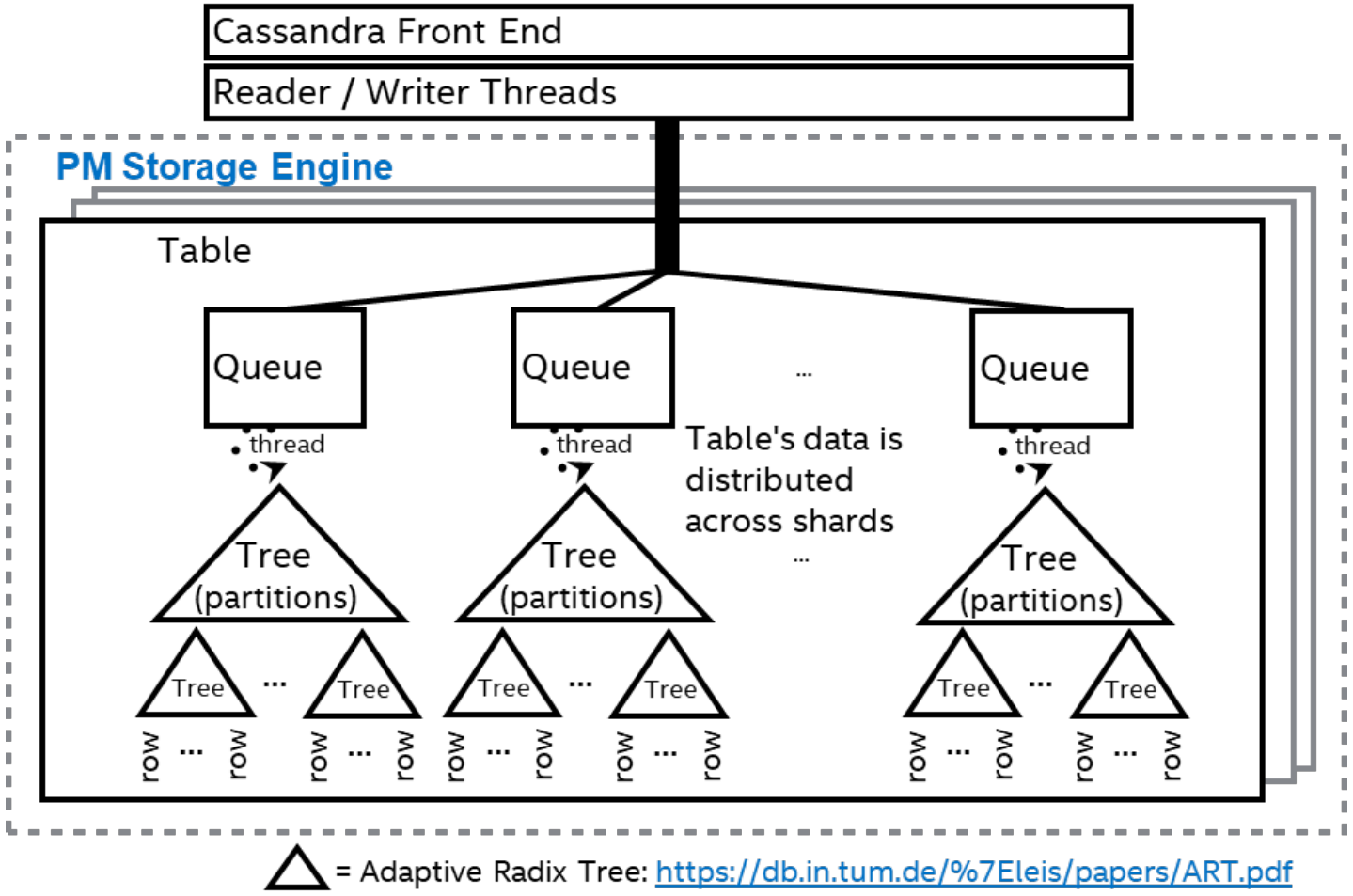


Alternate engines or mixture of engines at table granularity

Cassandra Persistent Memory Storage Engine

- Uses open source components
  - Low level Persistence Library
  - PMDK
- Adaptive Radix Tree
- Pluggable engine
- 6 - 8X speedups on reads and writes

https://github.com/intel/cassandra-pmem



Considerations on Persistent Use of Persistent Memory

- Data Integrity
  - On a controlled shutdown : flush caches
  - On an uncontrolled shutdown (e.g. power failure, crash): transactions
- Concurrency
  - CAS + flush is not atomic
- Fragmentation
  - Existing problem worsened by longer lived memory pools

Summary

- Persistent memory is available and valuable
  - Upstreamed in open source projects
- Multiple ways to extract value
  - No code change OR data structure redesign
- Libraries available to help
  - PMDK suite
  - Low Level Persistence Library - Java
  - Memory Mapped files

Links

Hbase Bucket Cache: https://issues.apache.org/jira/browse/HBASE-21874

MemcacheD: https://github.com/memcached/memcached/wiki/WarmRestart

Spark OAP: https://github.com/Intel-bigdata/OAP

Cassandra PMEM: https://github.com/intel/cassandra-pmem

PMDK: https://pmem.io/pmdk/

Libvmemcache: https://github.com/pmem/vmemcache

Low Level Persistence Library: https://github.com/pmem/llpl

Adaptive Radix Tree: https://db.in.tum.de/~leis/papers/ART.pdf