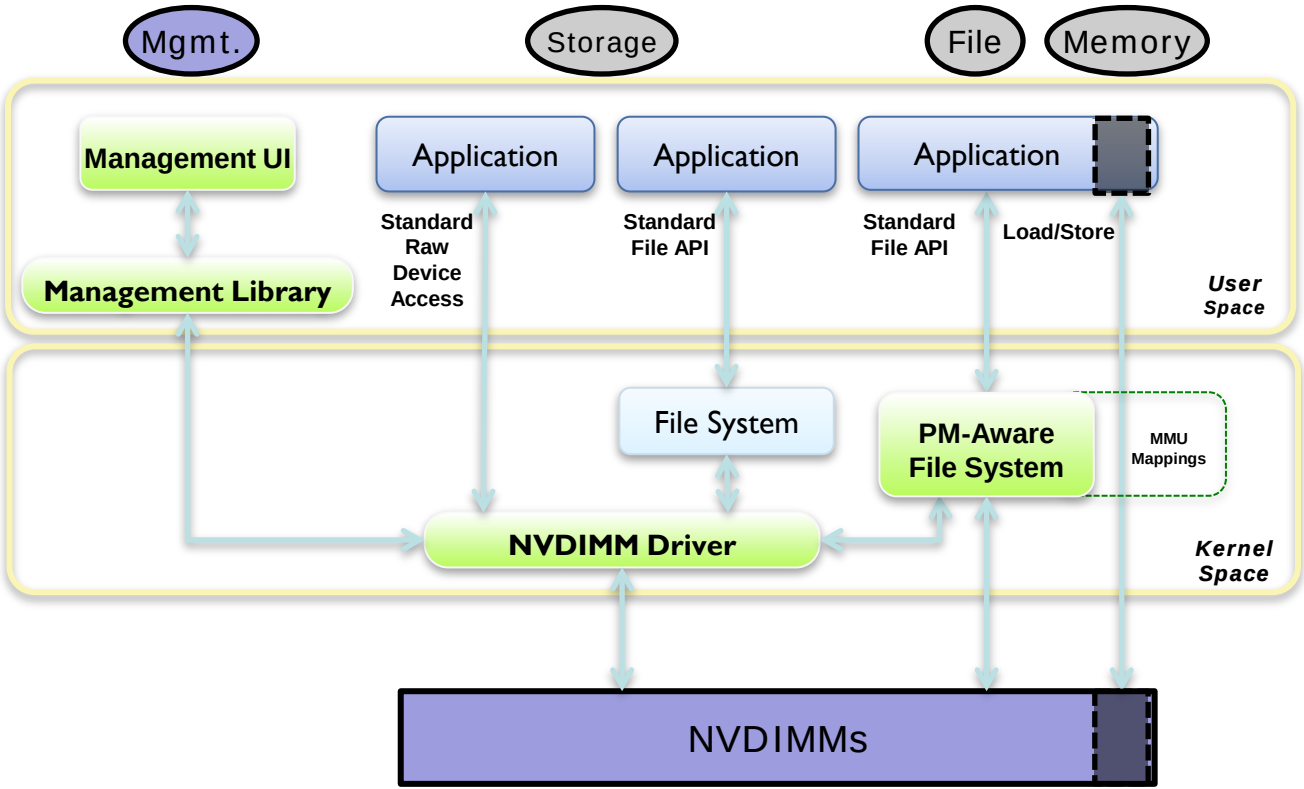
Persistent Memory Programming: The Current State and Future Direction

Andy Rudoff, Intel

Ancient History

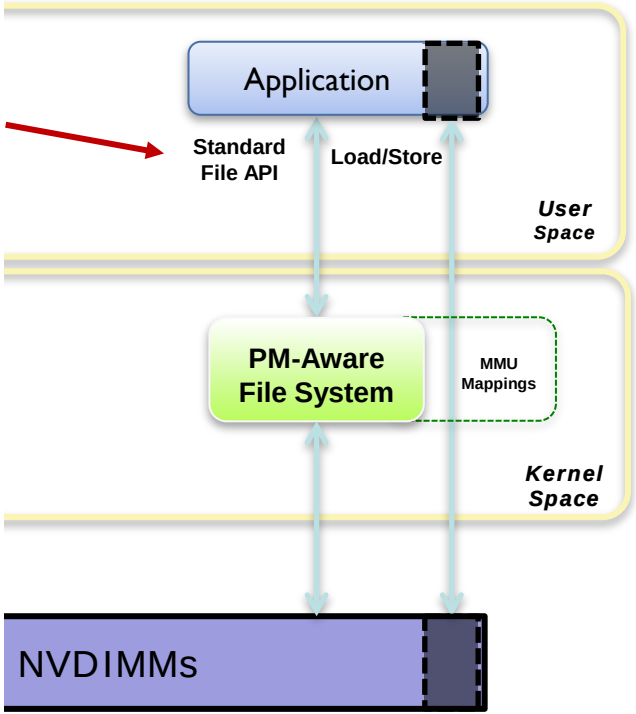
- June 2012
  - Formed the NVM Programming TWG
  - Immediate participation from key OSVs, ISVs, IHVs
- January 2013
  - Held the first PM Summit (actually called “NVM Summit”)
- January 2014
  - TWG published rev 1.0 of the NVM Programming Model

The Programming Model

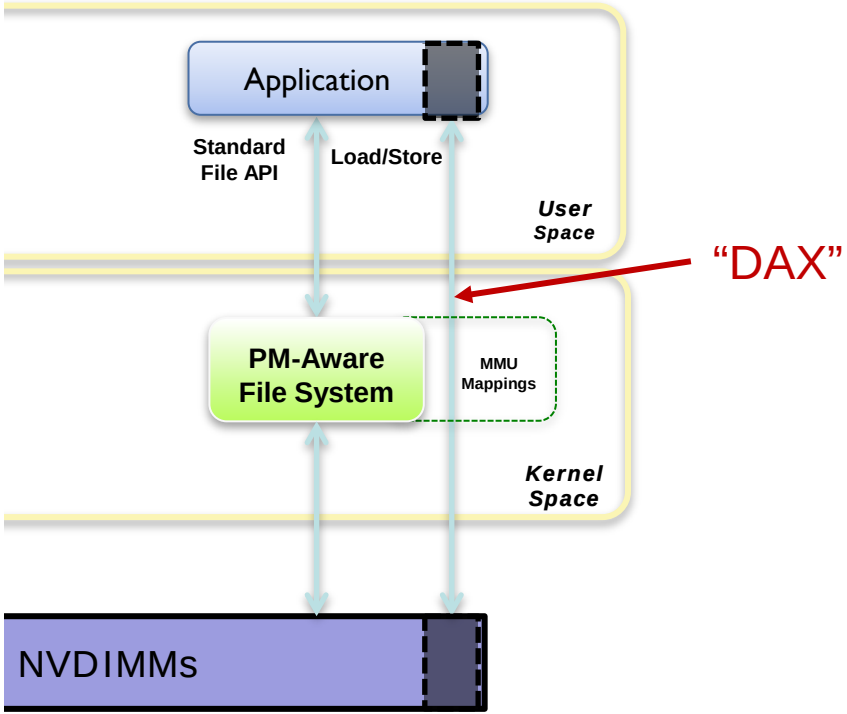


Must Open File Before Mapping

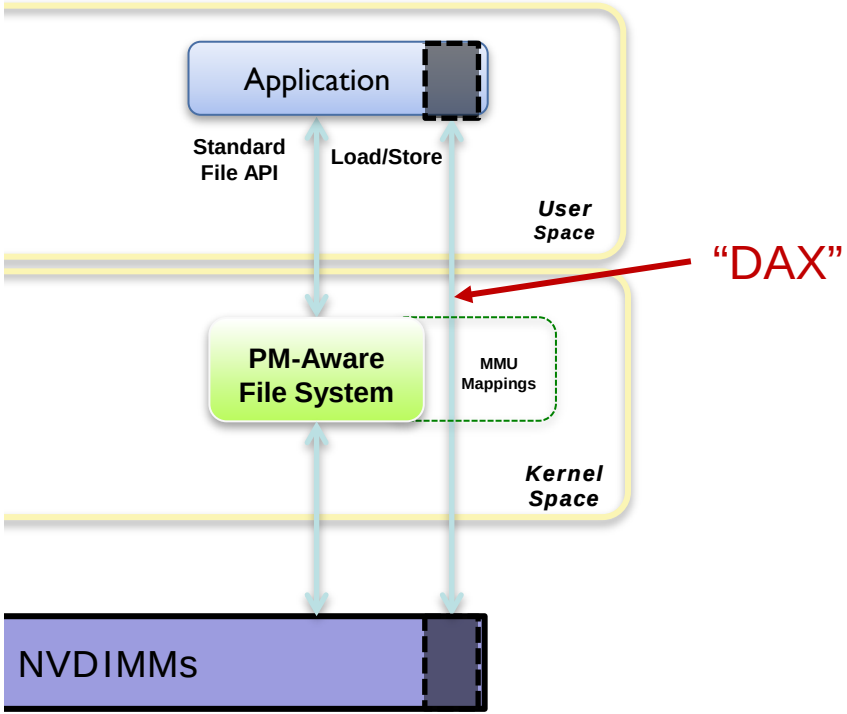
Standard Naming and Permission Model



Direct Access



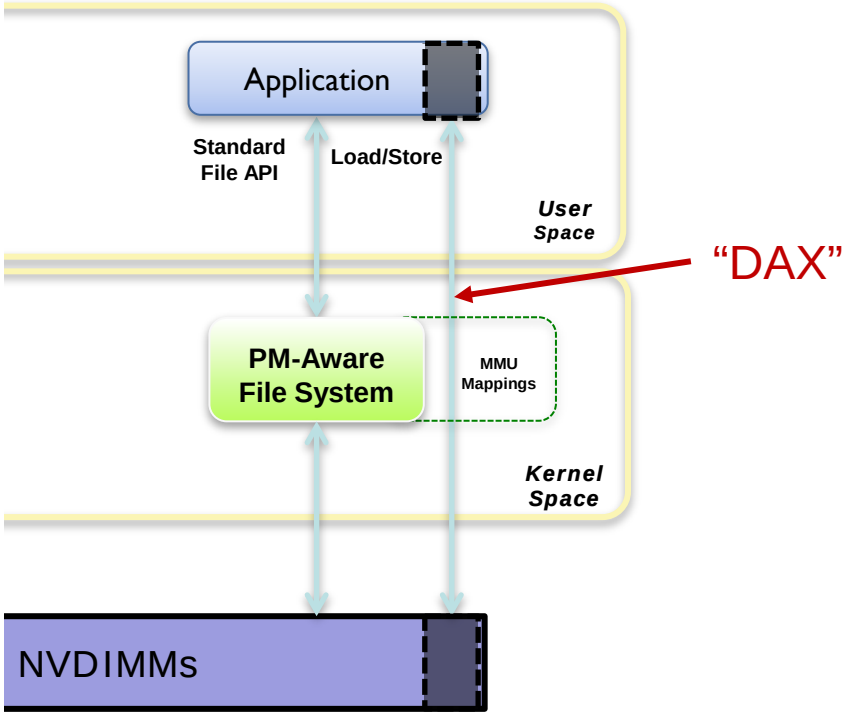
Direct Access



Windows:
- DAX Support is shipping
- NTFS is PM-Aware
- Some new APIs
- PMDK support

More info today from:
- Neal Christiansen
- Tom Talpey

Direct Access



Linux:

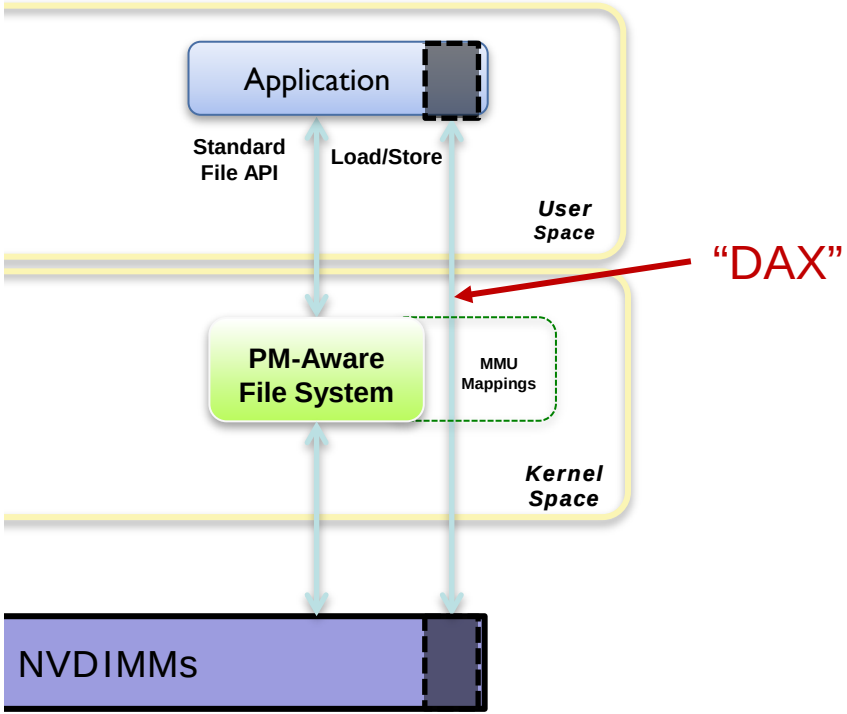
- DAX Support is shipping
- ext4 is PM-Aware
- XFS is PM-Aware
- PMDK support

- More filesystems coming

More info today from:

- Dan Williams
- Amit Golander

Direct Access



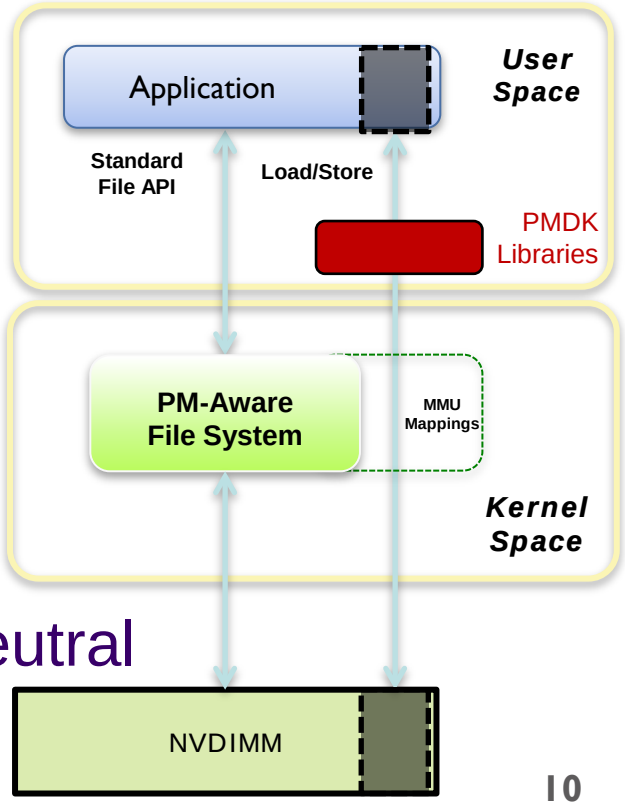
VMware:
Virtualization of PM

More info today from:
Rich Brunner

Applications: Public Demos

- 2017 was an interesting year for demos…
- SAP SAPPHIRE
- Oracle OpenWorld
- Built on the Persistent Memory programming model!

- PMDK Provides a Menu of Libraries
  - Developers pull in just what they need
    - Transaction APIs
    - Persistent memory allocators
  - Instead of re-inventing the wheel
    - PMDK libraries are fully validated
    - PMDK libraries are performance tuned
- PMDK Provides Tools for Developers
- PMDK is Open Source and Product-Neutral

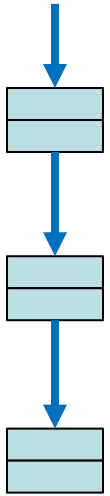


PMDK in a Nutshell

- Ten libraries, tools, examples…

```
void push(pool_base &pop, uint64_t value) {
        transaction::exec_tx(pop, [&] {
                auto n = make_persistent<pmem_entry>();

                n->value = value;
                n->next = nullptr;
                if (head == nullptr) {
                        head = tail = n;
                } else {
                        tail->next = n;
                        tail = n;
                }
        });
}
```

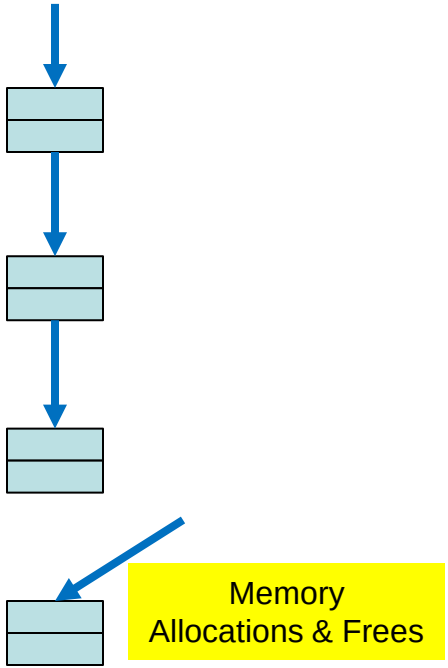


PMDK in a Nutshell

- Ten libraries, tools, examples…

```
void push(pool_base &pop, uint64_t value) {
        transaction::exec_tx(pop, [&] {
                auto n = make_persistent<pmem_entry>();

                n->value = value;
                n->next = nullptr;
                if (head == nullptr) {
                        head = tail = n;
                } else {
                        tail->next = n;
                        tail = n;
                }
        });
}
```



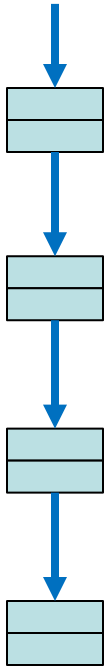
PMDK in a Nutshell

- Ten libraries, tools, examples…

```
void push(pool_base &pop, uint64_t value) {
        transaction::exec_tx(pop, [&] {
                auto n = make_persistent<pmem_entry>();

                n->value = value;
                n->next = nullptr;
                if (head == nullptr) {
                        head = tail = n;
                } else {
                        tail->next = n;
                        tail = n;
                }

        });
}
```



Multiple Operations Made Atomic

PMDK in a Nutshell

- Complex transactions, allocation handled by libraries
  - No “flush” calls to manage in most cases
  - Each ISV doesn’t have to re-invent
  - Performance tuned (esp for future enhancements)
- Licensing is very liberal
  - Steal all the code you want!
- PMDK is a convenience, not a requirement
  - Build your own library if you like!

Persistent Collections for Java

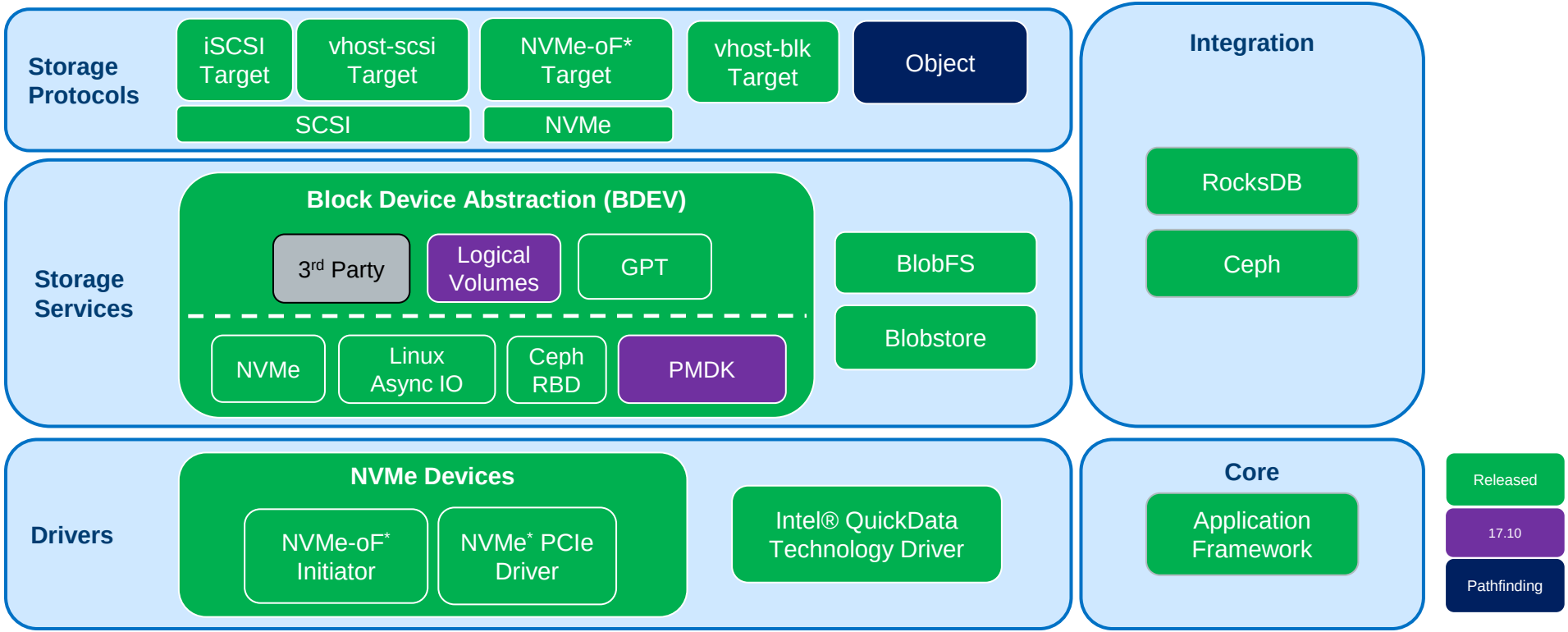

```
PersistentSortedMap employees = new PersistentSortedMap();
...
employees.put(id, data);
```

No flush calls.
Transactional.
Java library
handles it all.

See “pilot” project at: https://github.com/pmem/pcj

Storage Performance Developer Kit

spdk.io



TWG Ongoing Work

- Security
  - PM Hardware Security Threat Model (balloting)
- Remote persistent memory (via RDMA)
  - Ongoing – optimizations for RDMA worked in multiple forums
  - Remote asynchronous flush (under discussion)
- Higher-level Semantics
  - As we learn more..

More Information

- http://snia.org/PM
  - Specs, workgroups, webcasts, videos, presentations
- http://pmem.io
  - PMDK and other persistent memory programming information
- http://pmem.io/documents
  - Links to publications, standards, Windows & Linux info
- http://software.intel.com/pmem
  - Intel Developer Zone for persistent memory programming