14th ANNUAL WORKSHOP 2018

PERSISTENT MEMORY PROGRAMMING

Andy Rudoff

Intel

April 10, 2018

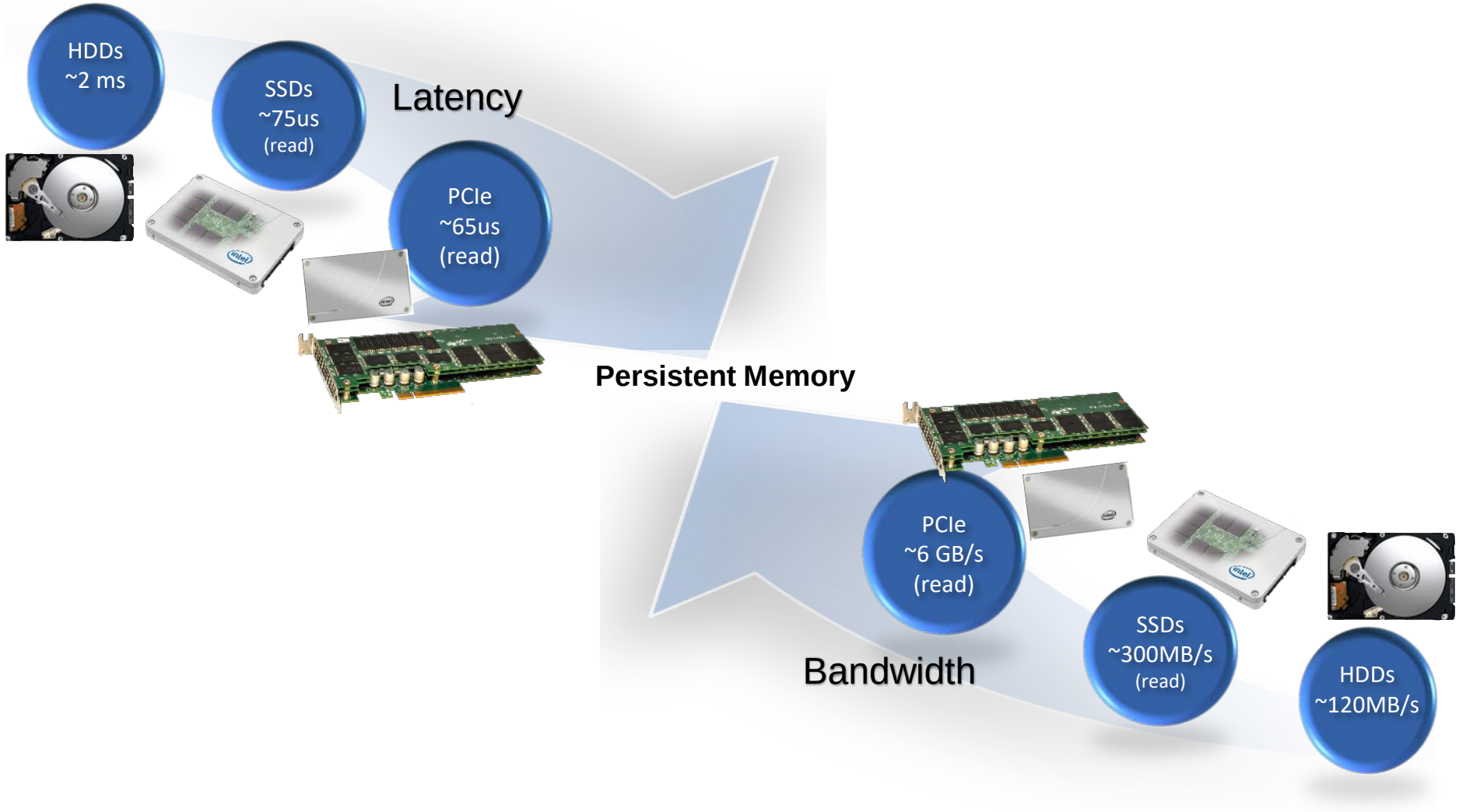
PERSISTENT MEMORY…

- **What is it?**
- **Why is it interesting?**
- **How does a program use it?**
- **What are the challenges?**
- **What's the state of the ecosystem?**

PERSISTENT MEMORY…

- **What is it?**
- **Why is it interesting?**
- **How does a program use it?**
- **What are the challenges?**
- **What's the state of the ecosystem?**

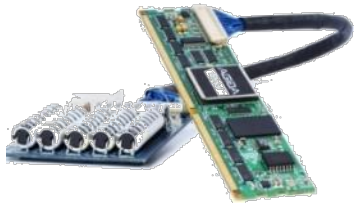
PROGRESSION OF STORAGE



DEFINITION OF PERSISTENT MEMORY

- **Byte-addressable**
  - As far as the programmer is concerned
- **Load/Store access**
  - Not demand-paged
- **Memory-like performance**
  - **Would reasonably stall a CPU load waiting for pmem**
- **Probably DMA-able**
  - Including RDMA

- **For modeling, think: Battery-backed DRAM**

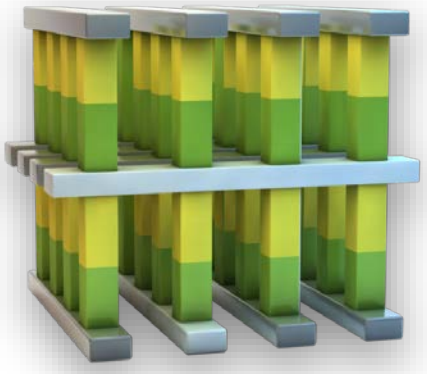


PERSISTENT MEMORY…

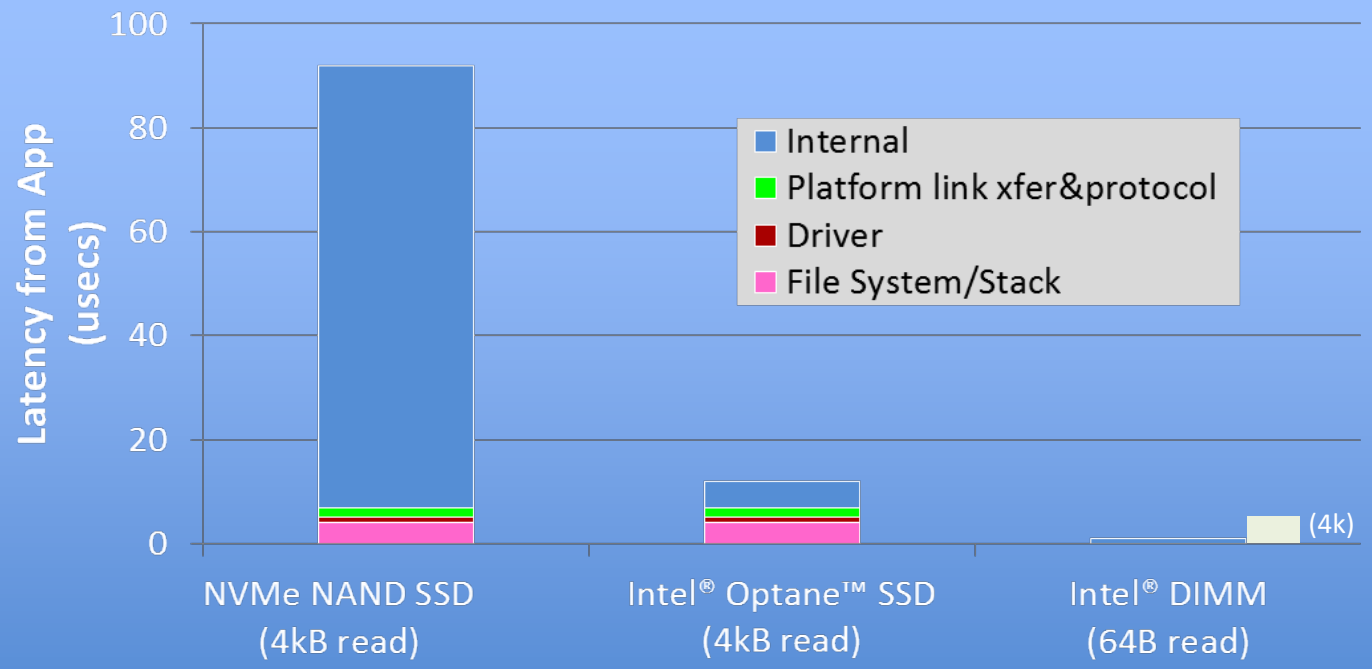
- What is it?
- **Why is it interesting?**
- How does a program use it?
- What are the challenges?
- What's the state of the ecosystem?

EMERGING TECHNOLOGY

- **3D XPoint™**
  - Persistent, Large Capacity & Byte Addressable
  - **6 TB** per two-socket system
  - DDR4 Socket Compatible
  - Can Co-exist with Conventional DDR4 DRAM DIMMs
  - Demonstrated at SAP Sapphire and Oracle Open World 2017
  - Cheaper than DRAM
  - Availability: 2018



OPTIMIZED SYSTEM INTERCONNECT



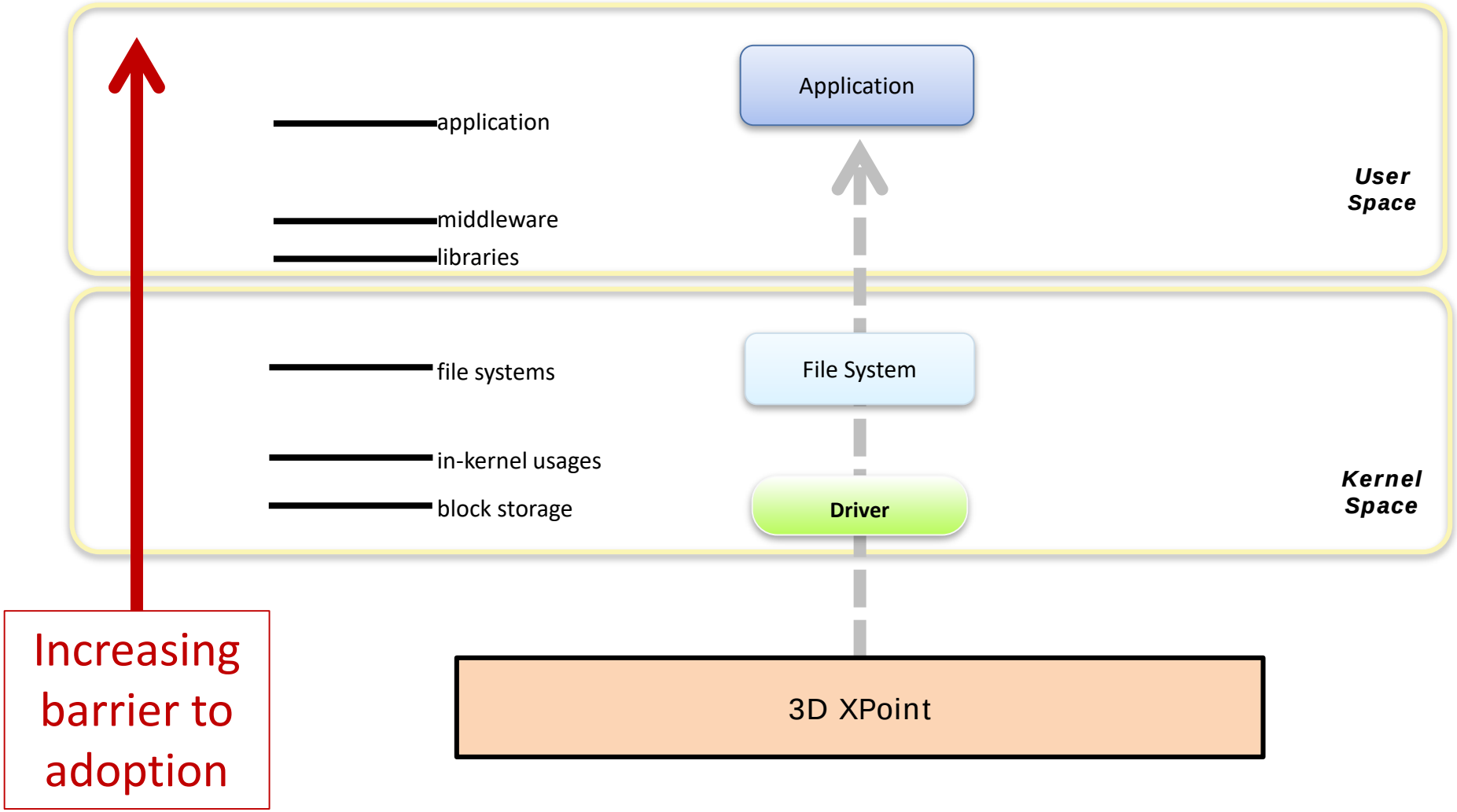
Reach full potential of 3D XPoint™ Technology by connecting it as Memory

Sources: "Storage as Fast as the rest of the system" 2016 IEEE 8th International Memory Workshop and measurement, Intel® Optane™ SSD measurements and Intel P3700 measurements, and technology projections

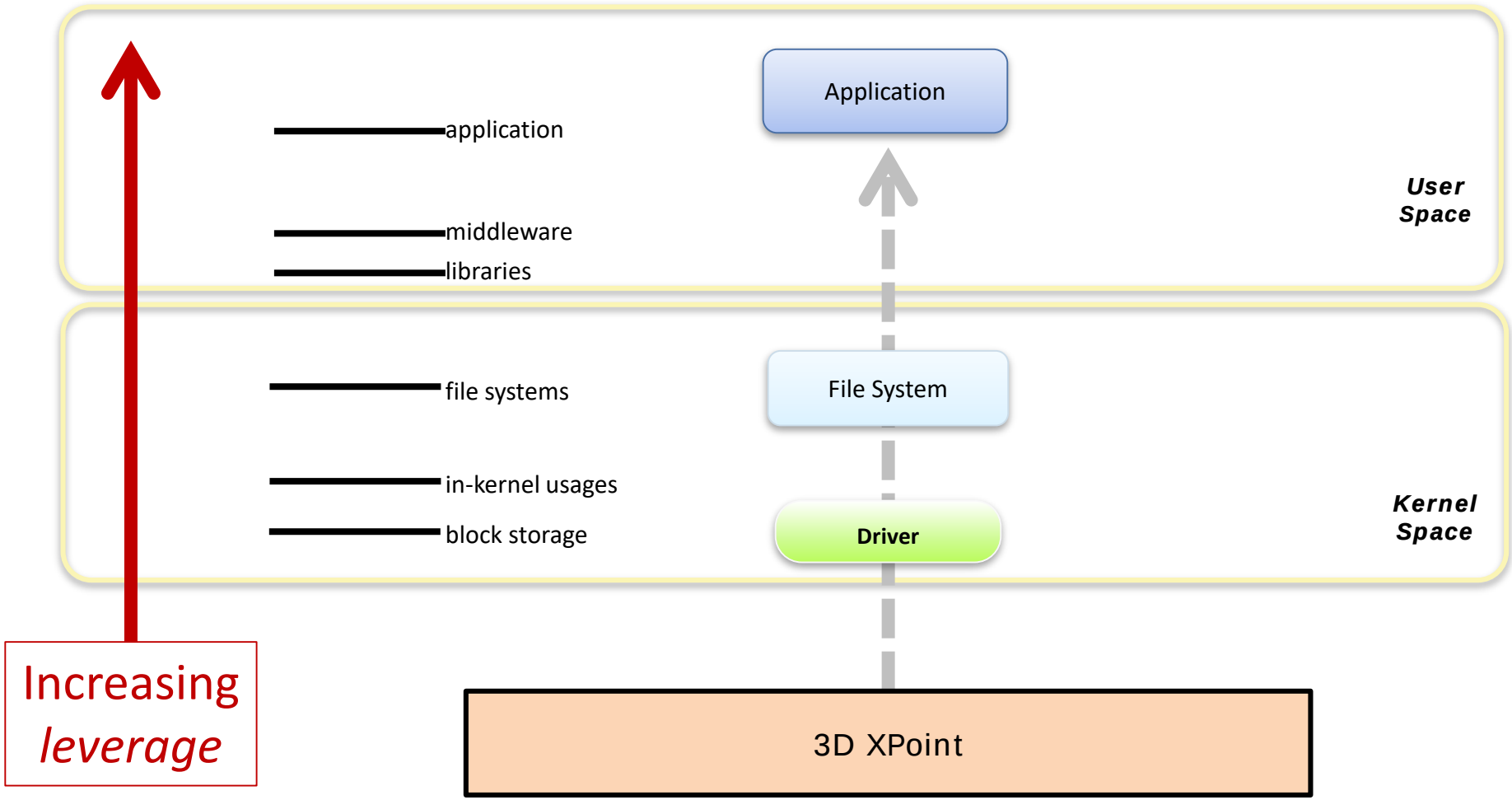
THE VALUE OF PERSISTENT MEMORY

- **Data sets addressable with no DRAM footprint**
  - At least, up to application if data copied to DRAM
- **Typically DMA (and RDMA) to pmem works as expected**
  - **RDMA directly to persistence – no buffer copy required!**
- **The "Warm Cache" effect**
  - No time spend loading up memory
- **Byte addressable**
- **Direct user-mode access**
  - No kernel code in data path

TRANSPARENCY LEVELS



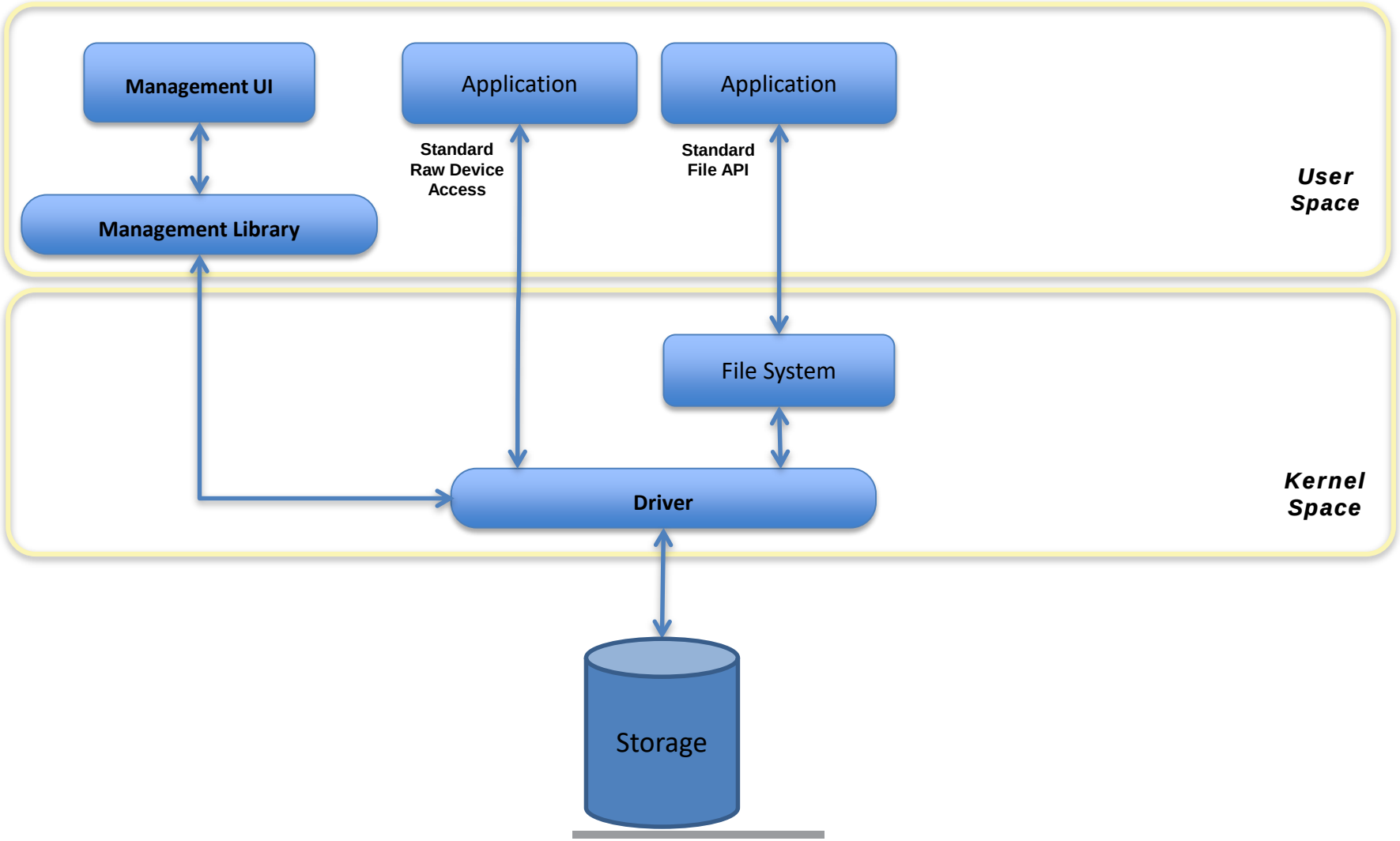
TRANSPARENCY LEVELS



PERSISTENT MEMORY…

- What is it?
- Why is it interesting?
- **How does a program use it?**
- What are the challenges?
- What's the state of the ecosystem?

THE STORAGE STACK (50,000 FOOT VIEW)



PROGRAMMER'S VIEW

```
fd = open("/my/file", O_RDWR);
…
count = read(fd, buf, bufsize);
…
count = write(fd, buf, bufsize);
…
close(fd);
```

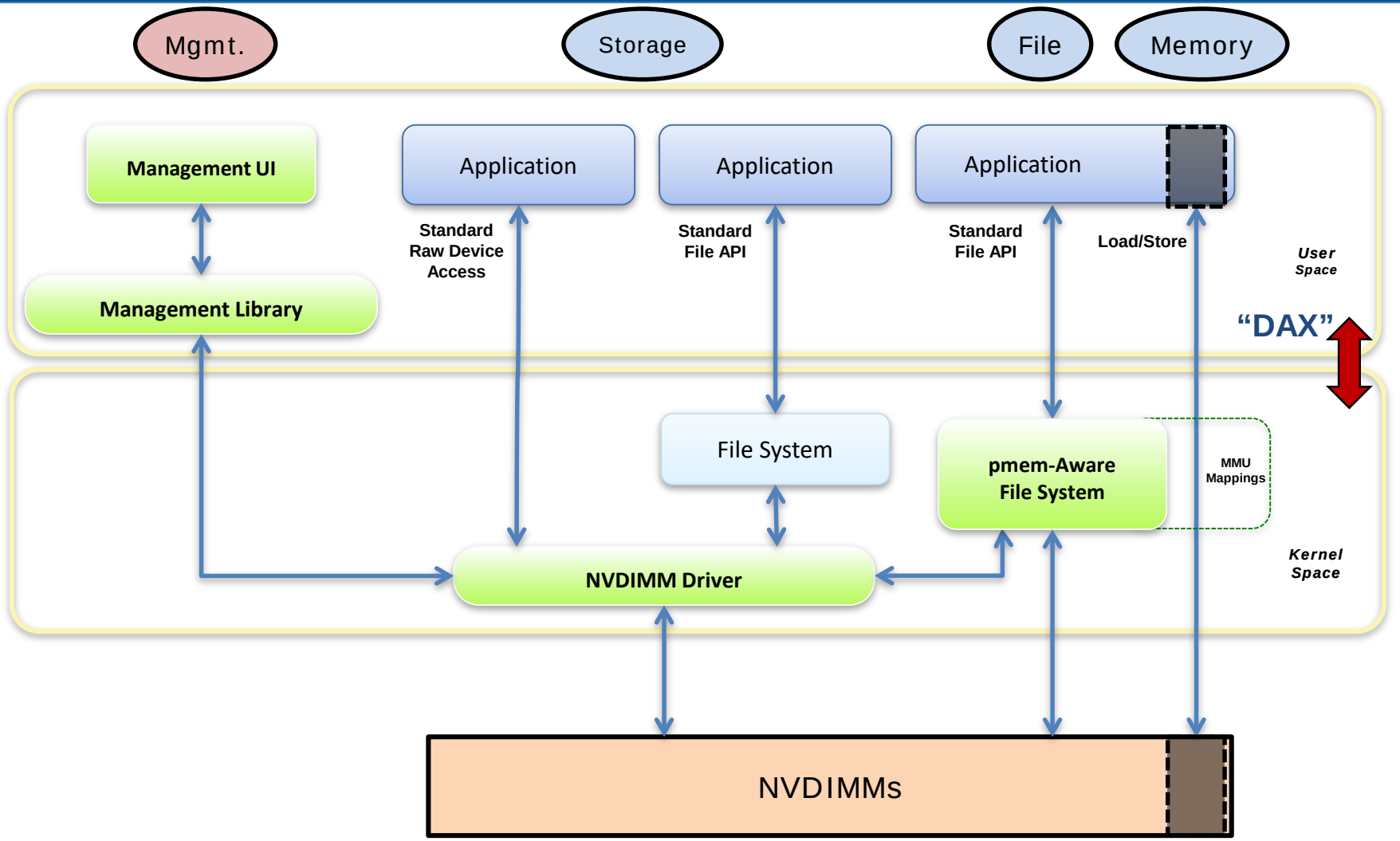
"Buffer-Based"

MEMORY-MAPPED FILES

```
fd = open(“/my/file”, O_RDWR);
…
base = mmap(NULL, filesize, PROT_READ|PROT_WRITE,
                        MAP_SHARED, fd, 0);
close(fd);
…
base[100] = ‘X’;
strcpy(base, “hello there”);
*structp = *base_structp;
…
```

“Load/Store”

THE PROGRAMMING MODEL



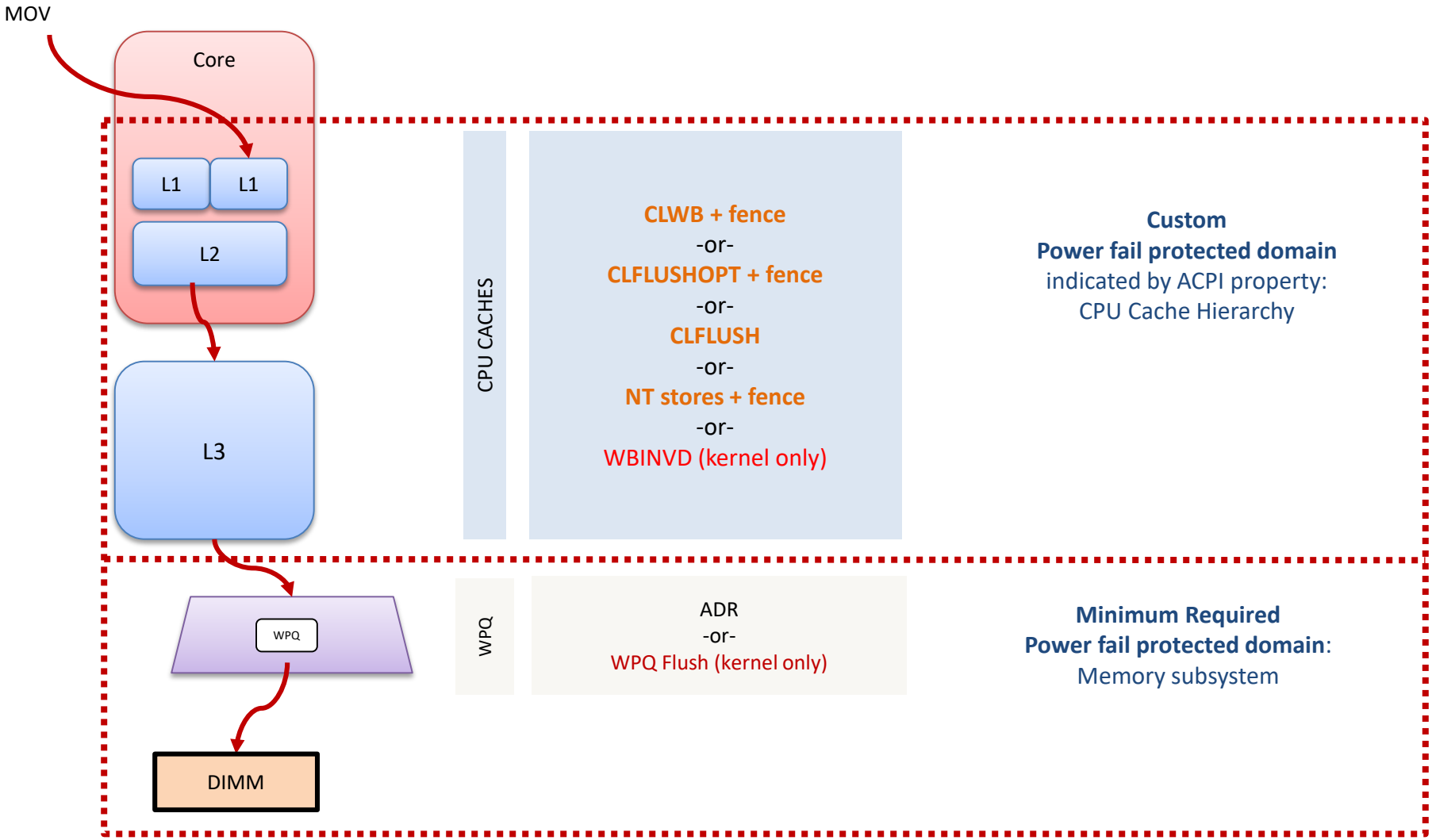
PERSISTENT MEMORY…

- What is it?
- Why is it interesting?
- How does a program use it?
- **What are the challenges?**
- What's the state of the ecosystem?

CHALLENGES

- **Allocation**
  - Like malloc/free, but persistent memory aware
- **Consistency across failure**
  - Memory-resident data structures, but transactional updates
- **DMA and RDMA**
  - “just works” if persistence doesn’t matter
  - Gets interesting when persistence matters
    - See Tom’s talk on this next

THE PLATFORM HARDWARE



PERSISTENT MEMORY…

- What is it?
- Why is it interesting?
- How does a program use it?
- What are the challenges?
- **What's the state of the ecosystem?**

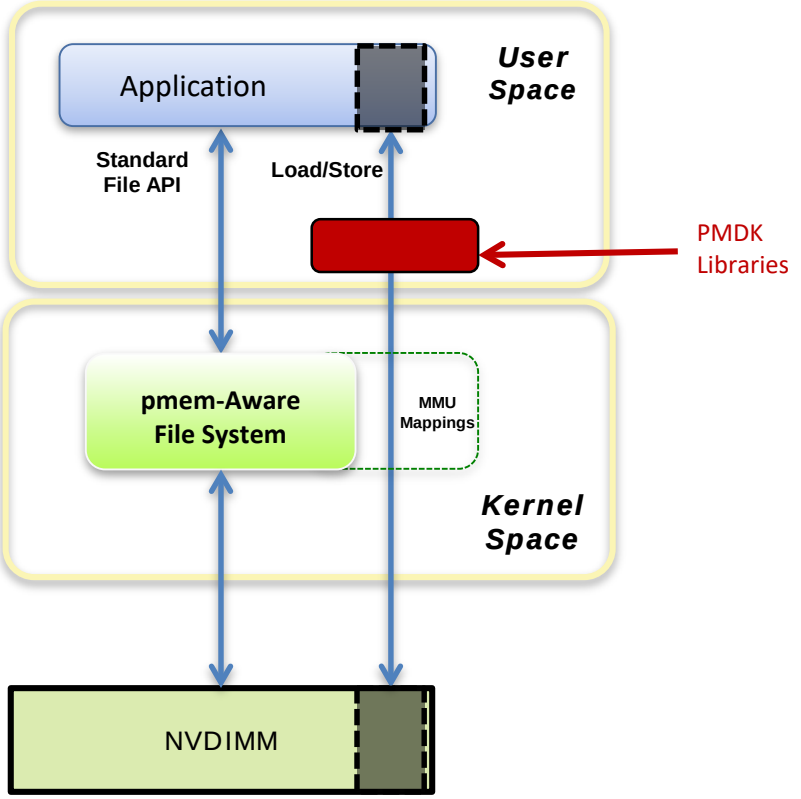
ECOSYSTEM

OS Detection of NVDIMMs	**ACPI 6.0+**
OS Exposes pmem to apps	***DAX* provides SNIA Programming Model** Fully supported: • Linux (ext4, XFS) • Windows (NTFS)
OS Supports Optimized Flush	Specified, but evolving (ask when safe) • Linux: **safe** with MAP_SYNC • Windows: **safe**
Remote Flush	Proposals under discussion (works today with extra round trip)
Deep Flush	In latest specification (SNIA NVMP and ACPI)
Transactions, Allocators	PMDK: http://pmem.io C, C++, Java (early access), Python (very early access)
Virtualization	All VMMs planning to support PM in guest (KVM changes upstream, Xen in review, others too…)

PERSISTENT MEMORY DEVELOPER KIT (PMDK)

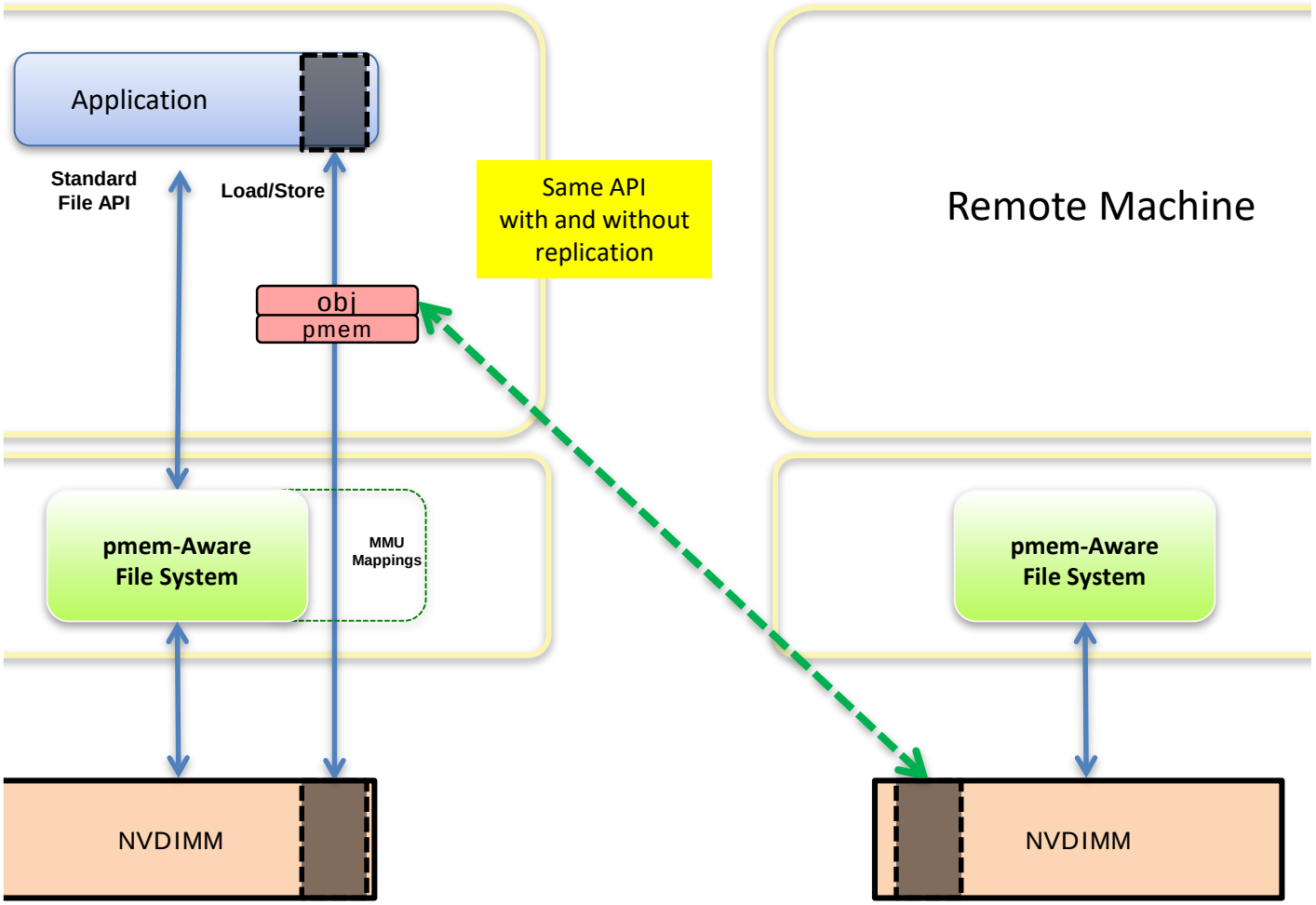
- http://pmem.io

- **PMDK Provides a Menu of Libraries**
  - Instead of re-inventing the wheel
    - PMDK libraries are fully validated
    - PMDK libraries are tuned for Intel hardware
    - Accelerates ISV readiness
  - Developers pull in just what they need
    - Transaction APIs
    - Persistent memory allocators



- **PMDK Provides Tools for Developers**
- **PMDK is Open Source and Product-Neutral**

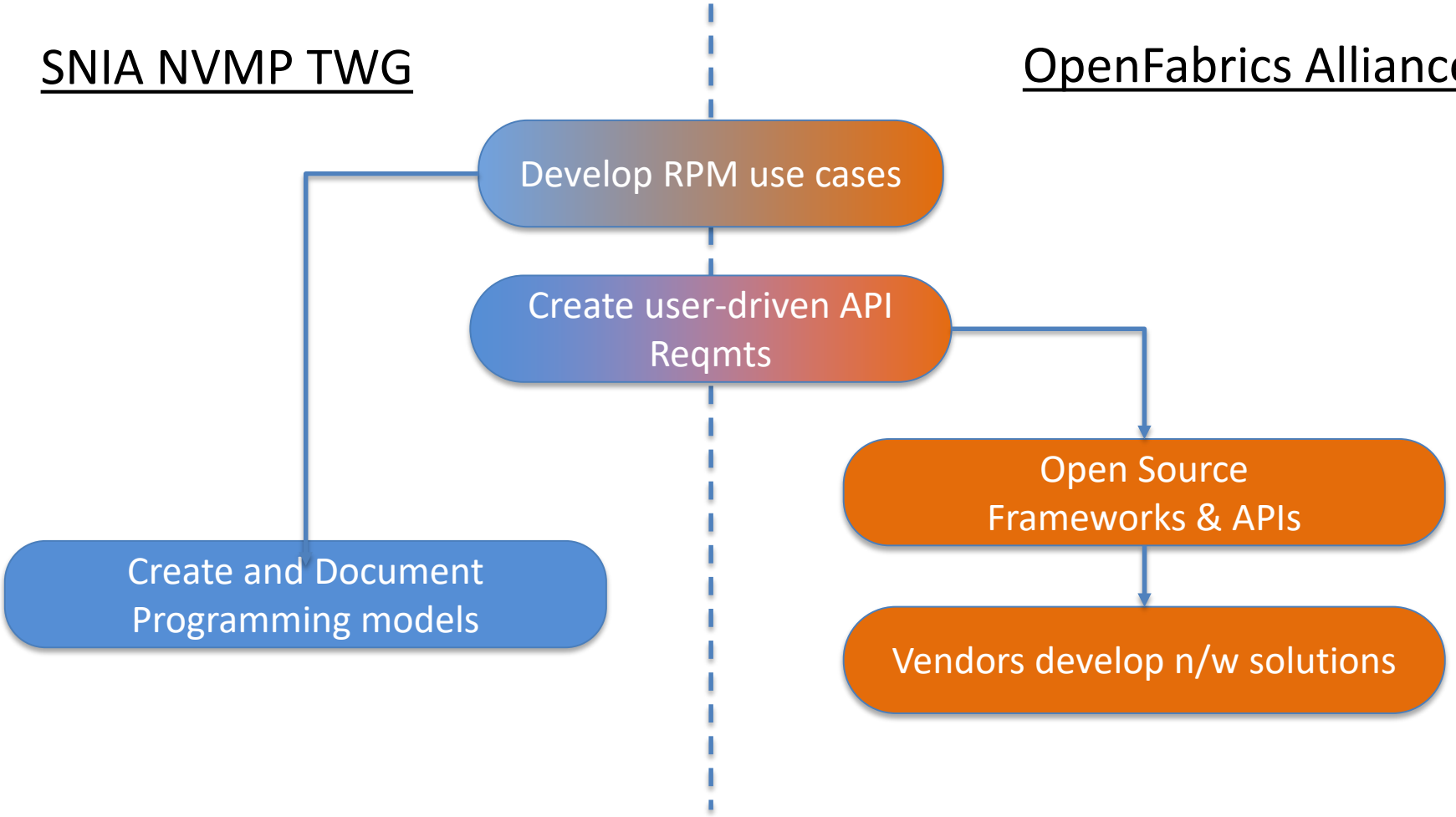
PMDK REPLICATION



SUMMARY

- **Persistent Memory technologies are emerging**
  - Some are available now
  - Some are available soon
  - Capacity explosion
- **The ecosystem has been preparing**
  - Pretty far along for local usage
  - Getting interesting for remote usages (Tom's talk)

ANNOUNCING - SNIA & OPENFABRICS ALLIANCE



14th ANNUAL WORKSHOP 2018

THANK YOU

Andy Rudoff

Intel