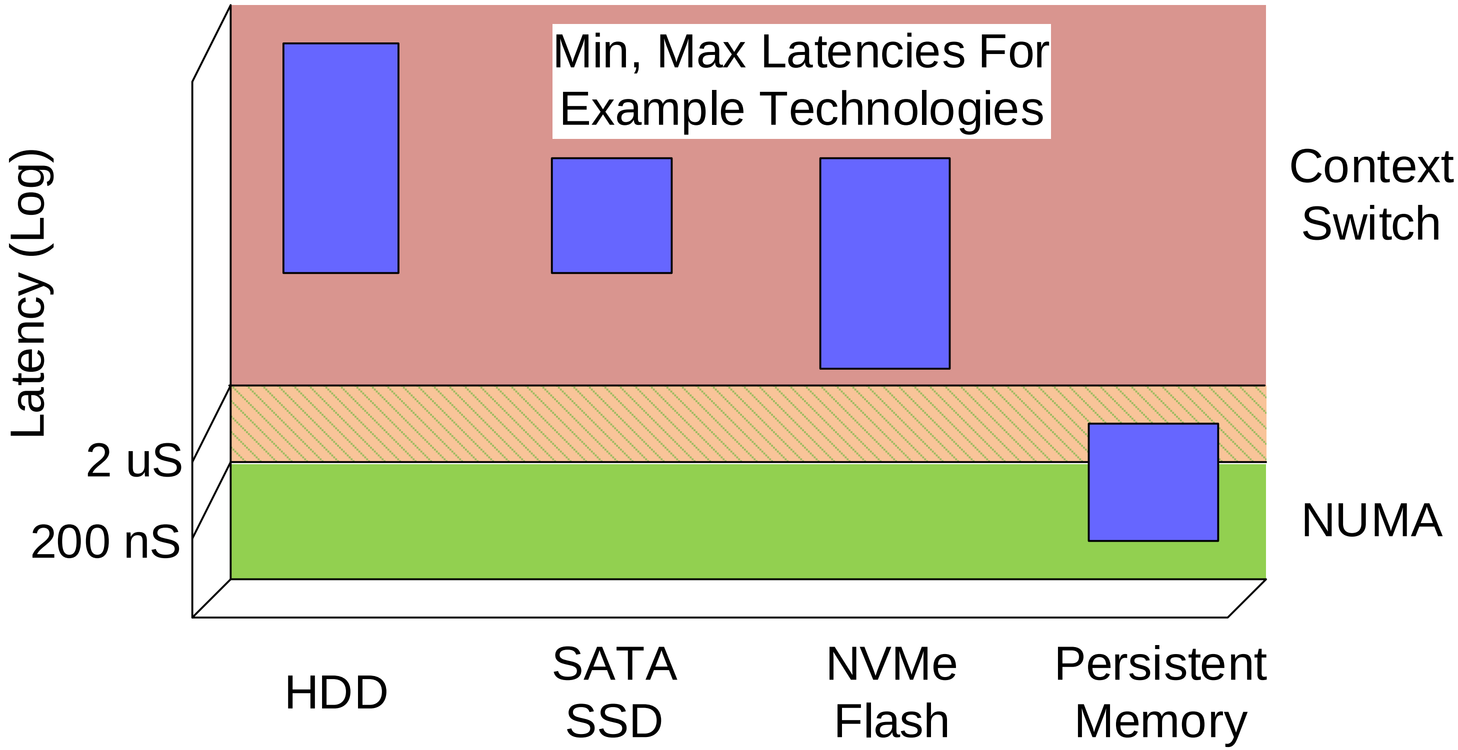
SDC
STORAGE DEVELOPER CONFERENCE
SNIA ■ SANTA CLARA, 2015

Preparing Applications for Persistent Memory

Doug Voigt
Hewlett Packard (Enterprise)

Latency Thresholds Cause Disruption



"Persistent memory" refers to memory-like non-volatile memory

SNIA NVM Programming Model

- Version 1.1 approved by SNIA in March 2015
  - http://www.snia.org/tech_activities/standards/curr_standards/npm
- Expose new block and file features to applications
  - Atomicity capability and granularity
  - Thin provisioning management
- Use of memory mapped files for persistent memory
  - Existing abstraction that can act as a bridge
  - Limits the scope of application re-invention
  - Open source implementations available
- Programming Model, not API
  - Described in terms of attributes, actions and use cases
  - Implementations map actions and attributes to API's

Persistent Memory Modes

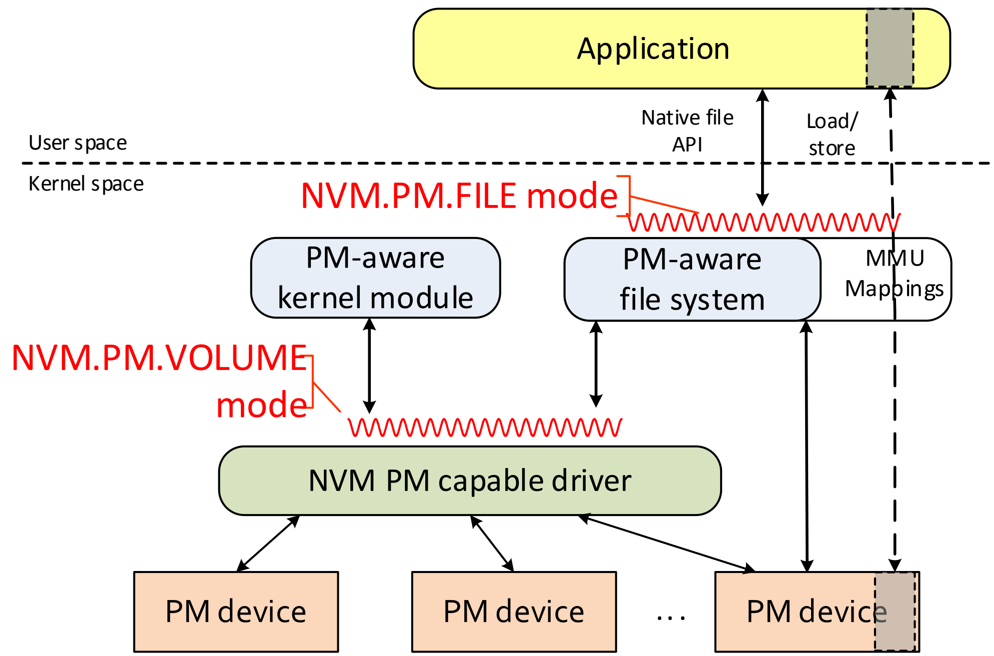
Use with memory-like NVM

NVM.PM.VOLUME Mode

- Software abstraction to OS components for Persistent Memory (PM) hardware
- List of physical address ranges for each PM volume
- Thin provisioning management

NVM.PM.FILE Mode

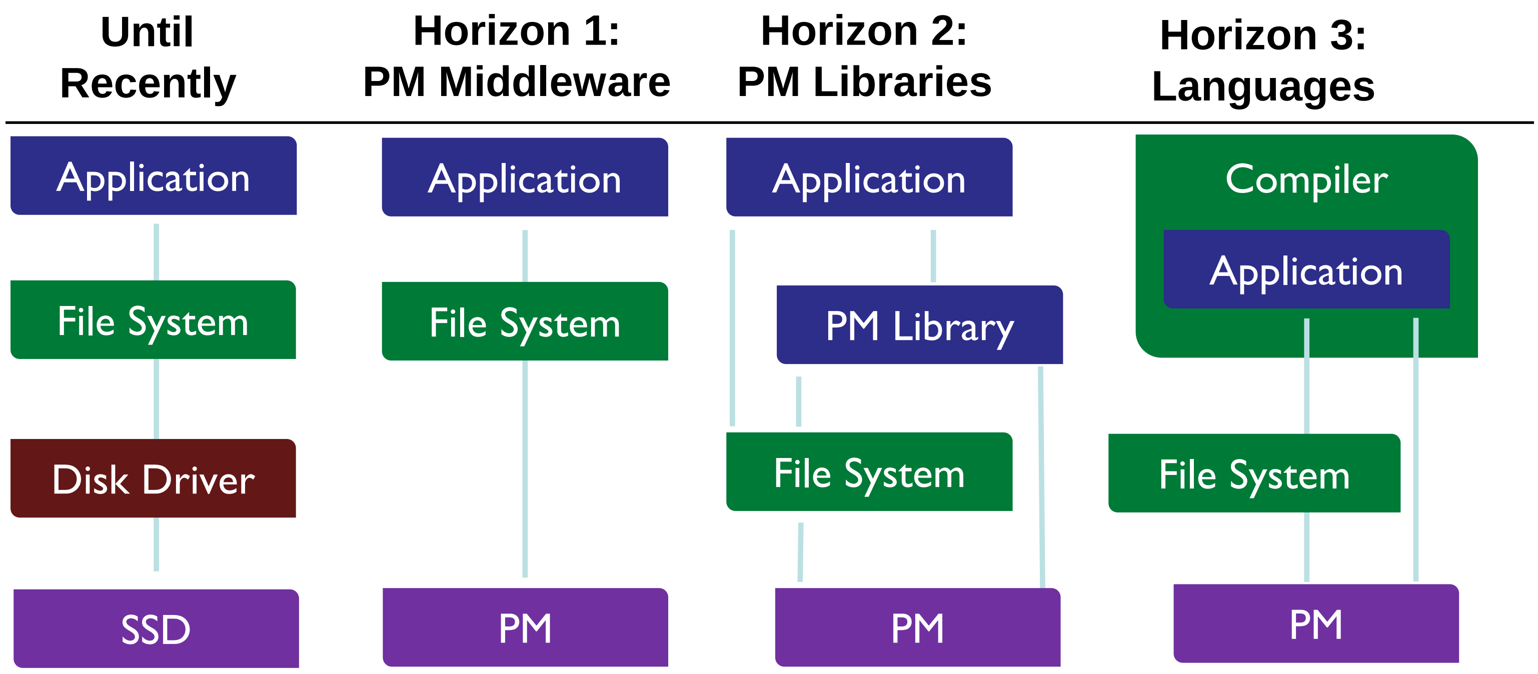
- Describes the behavior for applications accessing persistent memory Discovery and use of atomic write features
- Mapping PM files (or subsets of files) to virtual memory addresses
- Syncing portions of PM files to the persistence domain



Programming Model Application Impact

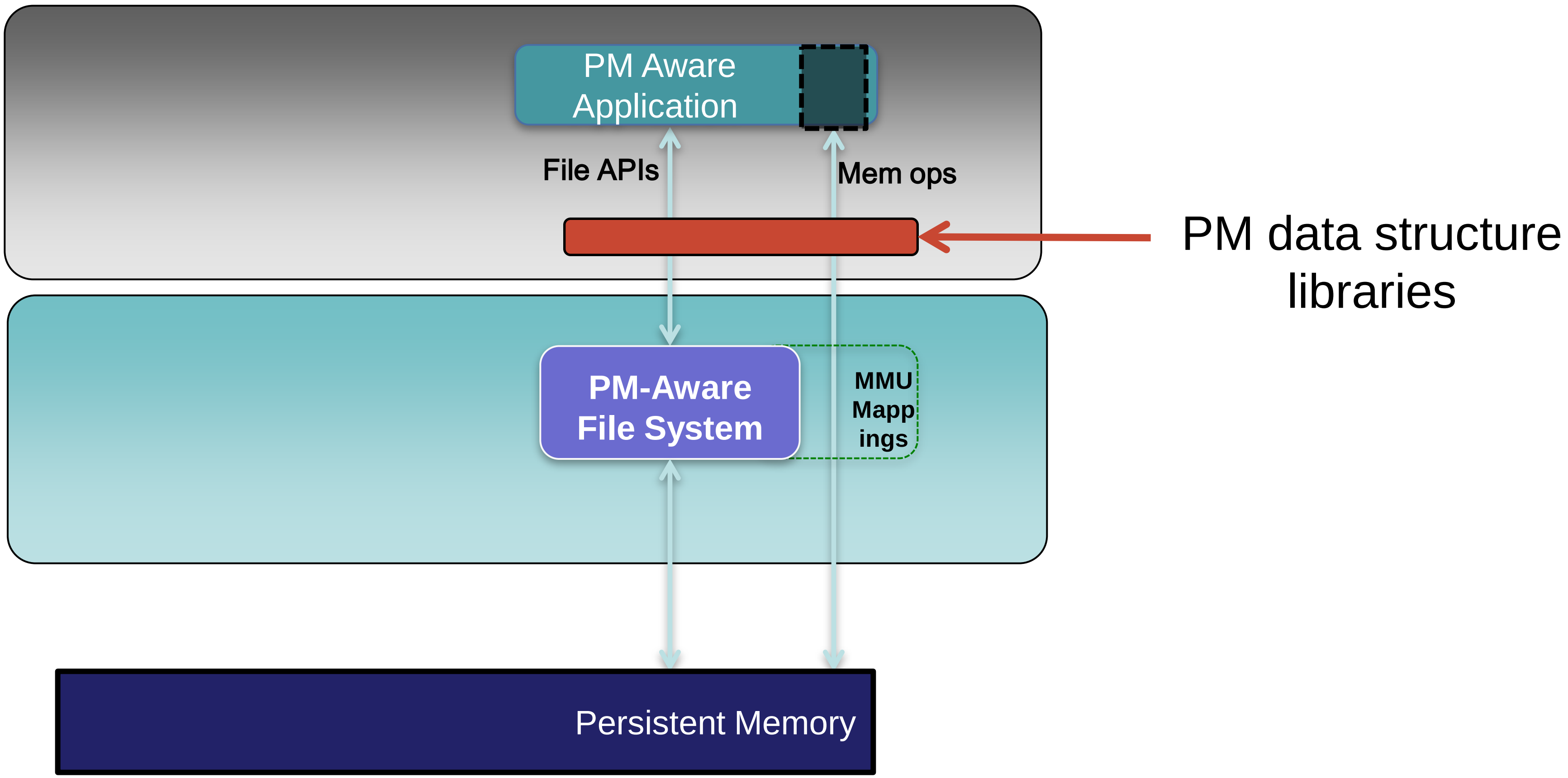
- Map and Sync Paradigm
  - Map associates memory addresses with PM in files
  - Sync ensures that modifications to data are persistent
  - Sync does not guarantee order
- Pointers – how do PM data structures reference each other?
  - Virtual addresses can be used as pointers?
  - Always use an offset from a re-locatable base?
- Failure Atomicity
  - Different from the inter-process consistency in current architectures
  - Processor architecture specific
- Exception Handling instead of status
  - If low level failure recovery fails
  - If backtracking is needed because PM was restored to an earlier state

Application Horizons

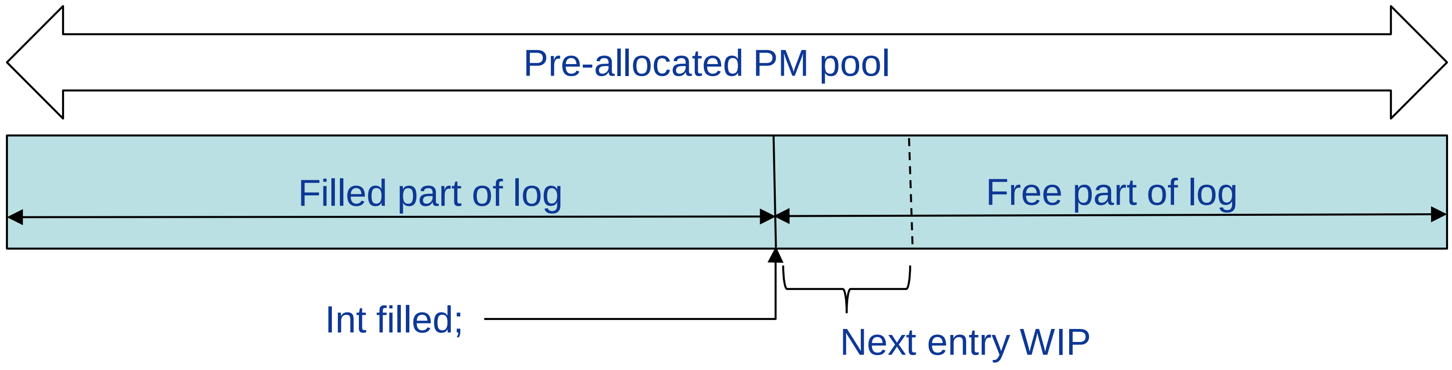


Persistent Memory Data Structures

Libraries Using NVM Programming Model



Trivial Example: Append Only Log



Append pseudo-code:

<Create new log entry in free space>

Sync(new entry);

filled = filled + size(new entry); # Atomic update to fundamental data type

Sync(filled);

PM Data Structures

- It can be more efficient to avoid modifying data in place
  - Use newly allocated space
  - PM allocation itself must be atomic/transactional
- Form groups of data structures
  - Within a PM pool
  - Cataloged under a common root
- Unify groups of PM data structures into larger transactions
  - Transaction object tracks and manages PM updates
  - Captures pre-images and rolls back if needed
  - Syncs/Flushes data to persistence domain

Pmem.io Library

- http://pmem.io/nvml
- PM assist functions
  - Map, Sync, Allocation
- PM Data Structures
  - Log, Block
- PM Object
  - Root, Transactions, Type Safety and more

Library vs. Language Extensions

- Features similar to pmem can be integrated into standard programming languages
  - More convenient
  - More sophisticated
  - Safer

http://www.hpl.hp.com/techreports/2013/HPL-2013-78.pdf

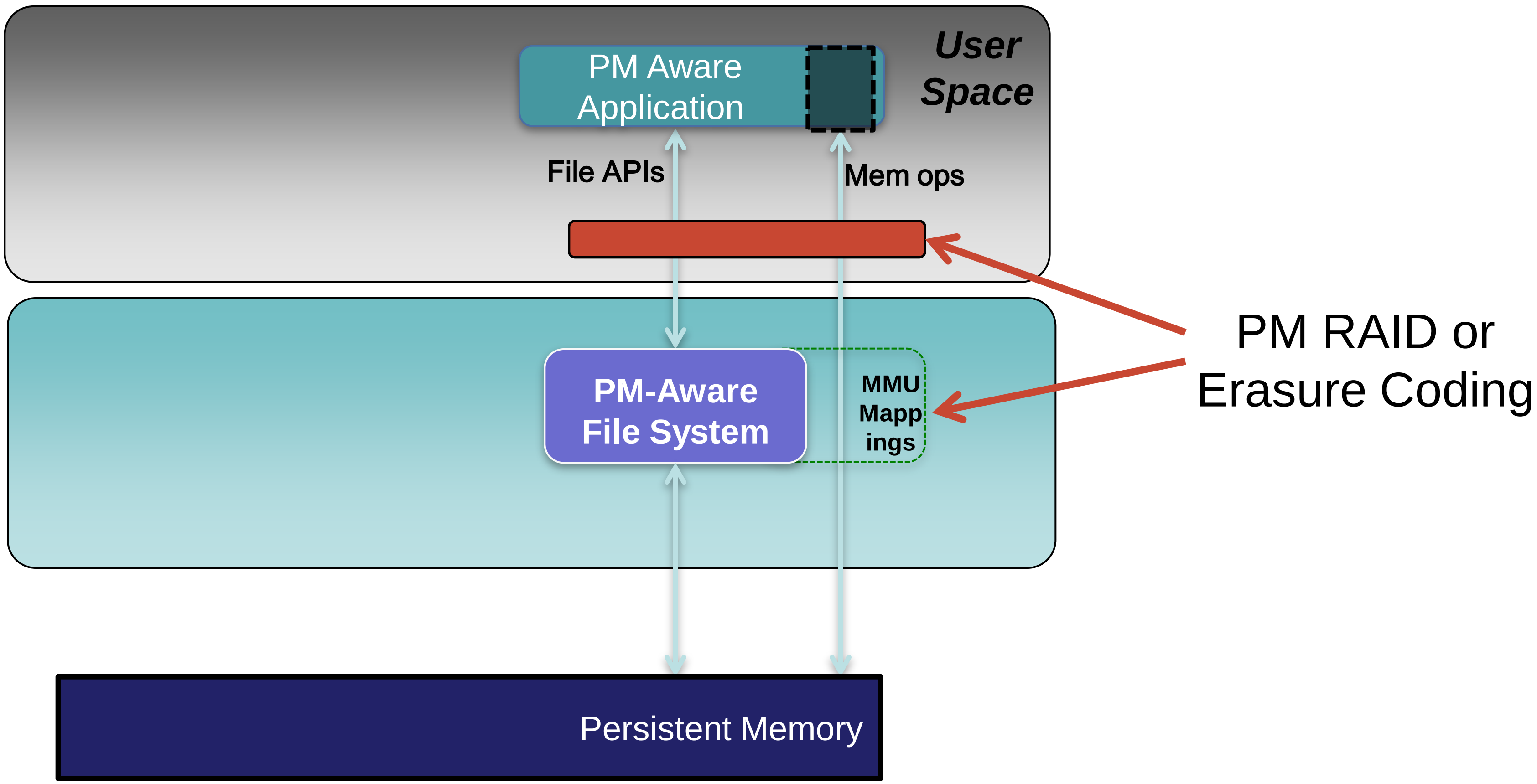
Failure atomic code sections based on existing critical sections

http://www.snia.org/sites/default/files/BillBridgeNVMSummit2015Slides.pdf

NVM region file management, transactions with locks, heap management

Failure Recovery

PM Fault Tolerance



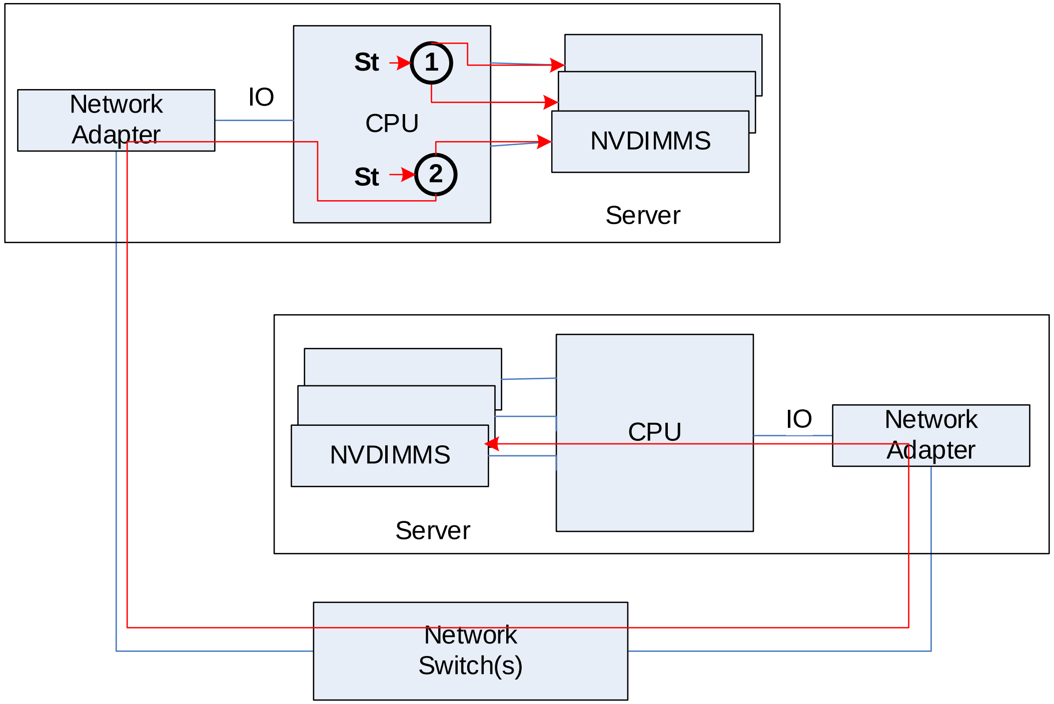
High Durability and High Availability (HA)

Durability

- Ability to (eventually) recover data after failure
- e.g. Local mirroring (1)
- Does not guarantee continuous access

Availability

- Ability to continuously access data regardless of failure
- Requires cross-node redundancy (2)
- High availability requires high durability



Remote Access for High Availability

- SNIA NVMP TWG work in progress
  - Use today's RDMA to explore this use case
  - Agnostic to specific implementation (IB, ROCE, iWARP)
  - Optimal implementation may not always be RDMA
- Recommends Remote OptimizedFlush network service
  - Goal is to minimize latency
  - Requires at least 2 round trips with today's implementations
  - Main issue is assurance of durability at remote site.
- New RDMA completion type helps
  - Proposed in Open Fabrics Alliance IO working group
  - Delays RDMA completion until data is in the remote persistence domain
  - Likely component of remote optimized flush implementation

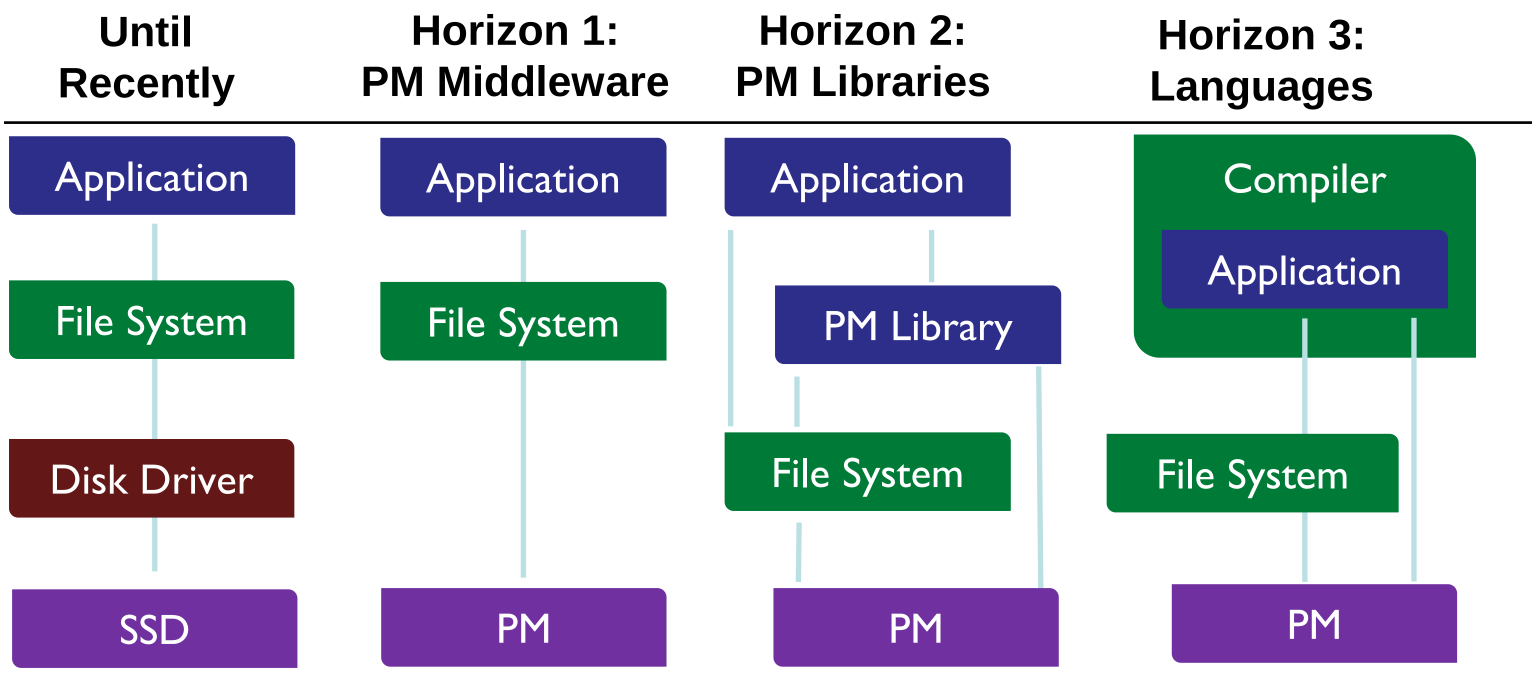
Application Recovery and Consistency

- Application level goal is recovery from failure
  - Requires robust local and remote error handling
  - High Availability (as opposed to High Durability) in today's systems requires application involvement.
  - High Availability is high latency (10's of uS) compared to memory
- Consistency is an application specific constraint
  - Uncertainty of data state after failure
  - Crash consistency
  - Higher order consistency points such as transactions
  - Atomicity of Aligned Fundamental Data Types
- Use consistency points to optimize HA performance
  - Periodic consistency points comprise groups of transactions
  - Apply recovery point objectives
  - Recovery may require application level backtracking

Backtracking Recovery

- Occurs when PM state is recovered to a recent consistency point
  - Created by remote optimized flush or transaction
  - Requires work in progress to be reconciled by the application
- Detection
  - During an exception
  - During a system or application restart
- Application Response
  - Transaction roll forward or roll back and retry
  - Consistency checking and correction

Application Horizons



Related Talks

Related Talks at SDC

- PM Hardware
  - The NVDIMM Cookbook: A Soup-to-Nuts Primer on Using NVDIMMs to Improve Your Storage Performance
    Jeff Chang, AgigA Tech and Arthur Sainio, Smart Modular
- PM Management
  - Managing the Next Generation Memory Subsystem
    Paul von Behren, Software Architect, Intel
- PM Performance
  - Load-Sto-Meter: Generating Workloads for Persistent Memory Doug Voigt, Damini Chopra, Storage CT Office, HP
- Remote Access and Failure Recovery
  - Remote Access to Ultra-low-latency Storage
    Tom Talpey, Architect, Microsoft
  - RDMA with PM: Software Mechanisms for Enabling Persistent Memory Replication, Chet Douglas, Principal SW Architect, Intel

Related Talks at SDC

- Applications of Persistent Memory
  - Solving the Challenges of Persistent Memory Programming
    Sarah Jelinek, Senior SW Engineer, Intel
  - Building NVRAM Subsystems in All-Flash Storage Arrays
    Pete Kirkpatrick, Principal Engineer, Pure Storage
- Keynote earlier today
  - Planning for the Next Decade of NVM Programming
    Andy Rudoff, SNIA NVM Programming TWG, Intel
- Also check out persistent memory presentations in the pre-conference
  - Advances in Non-Volatile Storage Technologies
  - Nonvolatile Memory (NVM), Four Trends in the Modern Data Center, and the Implications for the Design of Next Generation Distributed Storage Platforms
  - Developing Software for Persistent Memory

SDC

STORAGE DEVELOPER CONFERENCE

SNIA ■ SANTA CLARA, 2015

Preparing Applications for Persistent Memory

Doug Voigt
Hewlett Packard (Enterprise)