

Emerging Storage Interfaces: DAOS and PMDK

Adrian Jackson

Johann Lombardi

Mohamad Chaarawi

TUTORIAL

Emerging Storage Interfaces: DAOS and PMDK

Johann Lombardi
Mohamad Chaarawi



Adrian Jackson
(a.jackson@epcc.ed.ac.uk)



Aims

- Understand persistent memory hardware and software
- Learn how to program persistent memory
- Learn about DAOS
- Learn about DAOS apis and exploiting DAOS from your applications
- Get hands on with DAOS and persistent memory hardware

Aims cont.

- Understand data movement and think about application data requirements
- Thinking about different ways you undertake I/O or storing data
- Move beyond bulk, block-based, I/O paradigms

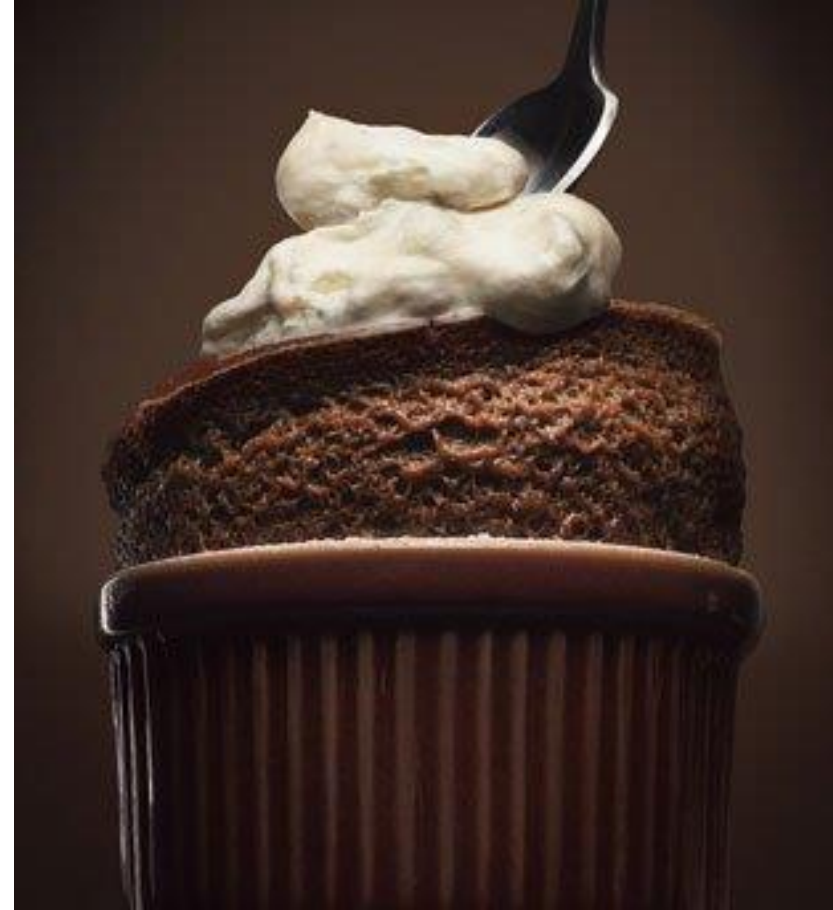
Format

- Lectures and practicals
- Slides and exercise material available online:
 - <https://github.com/NGIOproject/PMTutorial>
 - Exercises will be done on remote machine (NEXTGenIO system)
 - We will provide accounts for these

Timetable

- 08.30 Introduction
- 08.45 Persistent memory hardware and programming
- 09.00 Practical: Benchmarking persistent memory
- 09.30 Overview of the DAOS object store and CXL integration
- 10.00 Break
- 10.30 DAOS API and programming object stores
- 11.00 Demonstration: Exploiting DAOS for applications
- 11.30 DAOS storage interfaces
- 12.00 Summary and finish

Emerging storage interfaces



Programming persistent memory

```
double *a, *b, *c;
pmemaddr = pmem_map_file(path, array_length,
                        PMEM_FILE_CREATE|PMEM_FILE_EXCL,
                        0666, &mapped_len, &is_pmem)

a = pmemaddr;
b = pmemaddr + (*array_size+OFFSET)*BytesPerWord;
c = pmemaddr + (*array_size+OFFSET)*BytesPerWord*2;

#pragma omp parallel for
for (j=0; j<*array_size; j++){
    a[j] = b[j]+scalar*c[j];
}

pmem_persist(a, *array_size*BytesPerWord);
```


Using DAOS

```
mkdir /tmp/my_filesystem  
dfuse -m /tmp/my_filesystem --pool tutorial --cont adrians  
  
...  
  
fusermount3 -u /tmp/my_filesystem
```

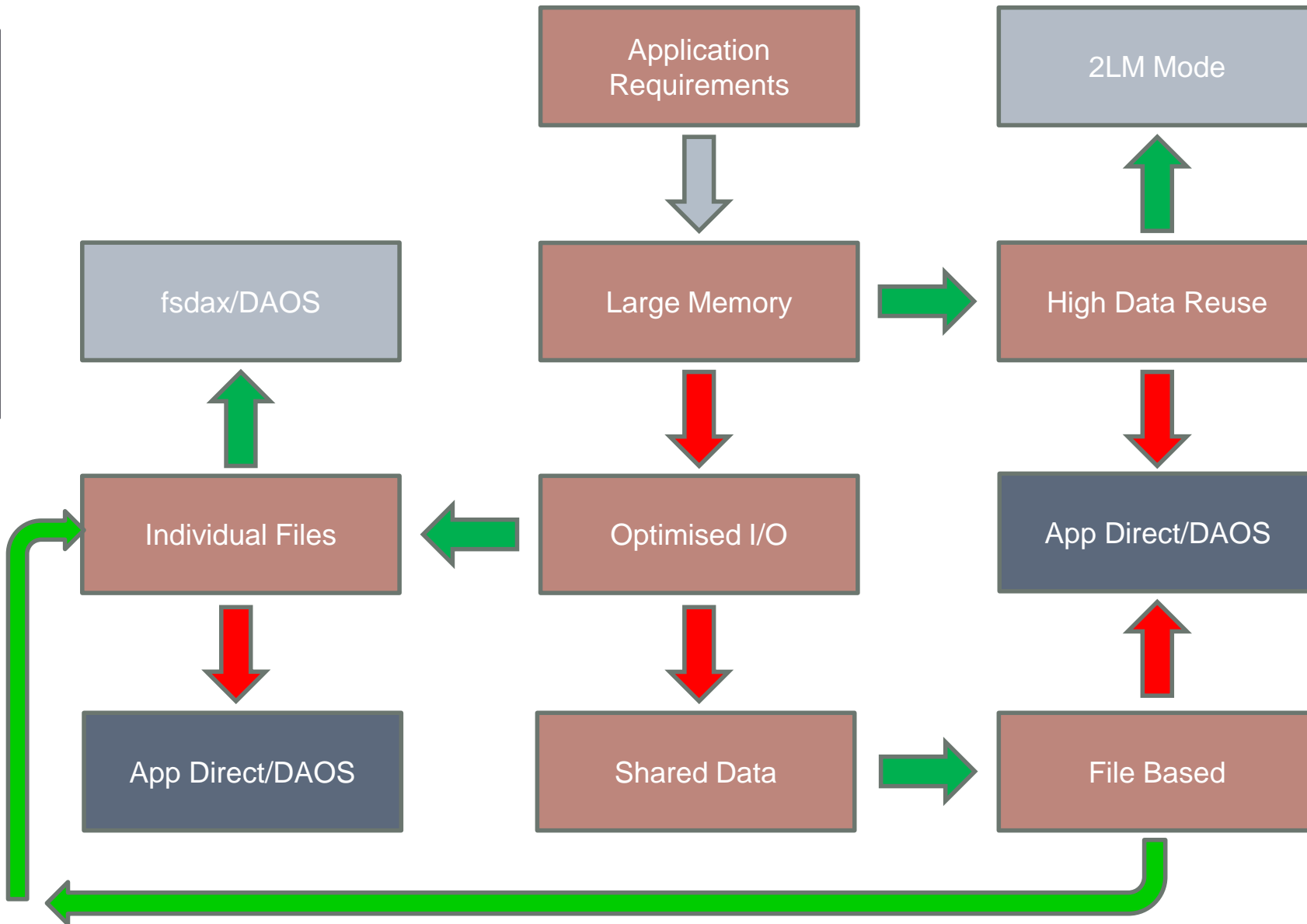
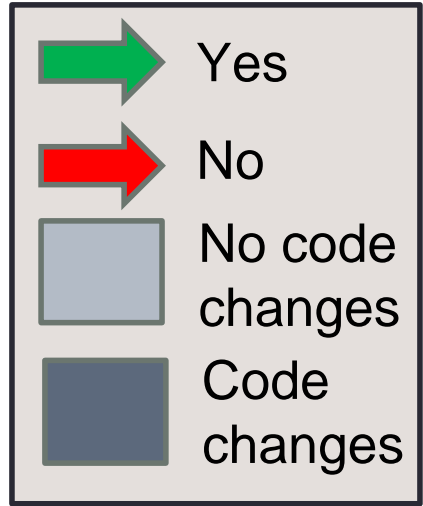
Programming DAOS

```
rc = daos_init();
daos_pool_info_t pool_info;
daos_cont_info_t co_info;
rc = daos_pool_connect(o->pool, o->group, DAOS_PC_RW, &poh,
&pool_info, NULL);
rc = daos_cont_open(poh, o->cont, DAOS_COO_RW, &coh, &co_info, NULL);
rc = dfs_mount(poh, coh, O_RDWR, &dfs);
rc = dfs_write(dfs, obj, &sgl, off, NULL);
rc = dfs_read(dfs, obj, &sgl, off, &ret, NULL);
rc = dfs_umount(dfs);
rc = daos_cont_close(coh, NULL);
rc = daos_cont_destroy(poh, o->cont, 1, NULL);
rc = daos_pool_disconnect(poh, NULL);
rc = daos_fini();
```

Programming persistent memory



Programming Persistent Memory



Programming persistent memory

- Design and performance considerations are the challenge
 - Programming the memory is easy
- Design for functionality
 - What is persistent, when is it persistent, what failures can you tolerate, etc..
- Design for performance
 - Memory size, I/O, data access costs, etc...
- Design for hardware configurations
 - NUMA, filesystems, storage, etc...

Object storage

- Data stored in unstructured objects
 - Data has identifier
 - Size and shape can vary
 - Metadata can also vary
- Originally designed for unstructured data sets
 - Bunch of data with no specific hierarchy required
- Can also enable efficient/fast access to data in different structures
 - Supports different creation, querying, analysis, and use patterns

Summary

- Please don't hesitate to ask questions!
- We have a practical session and a demonstration session
 - Login account will remain active for your to try out using PMDK and DAOS after the tutorial
 - Email a.jackson@epcc.ed.ac.uk to get an account on the system we will use for practical/try out sessions