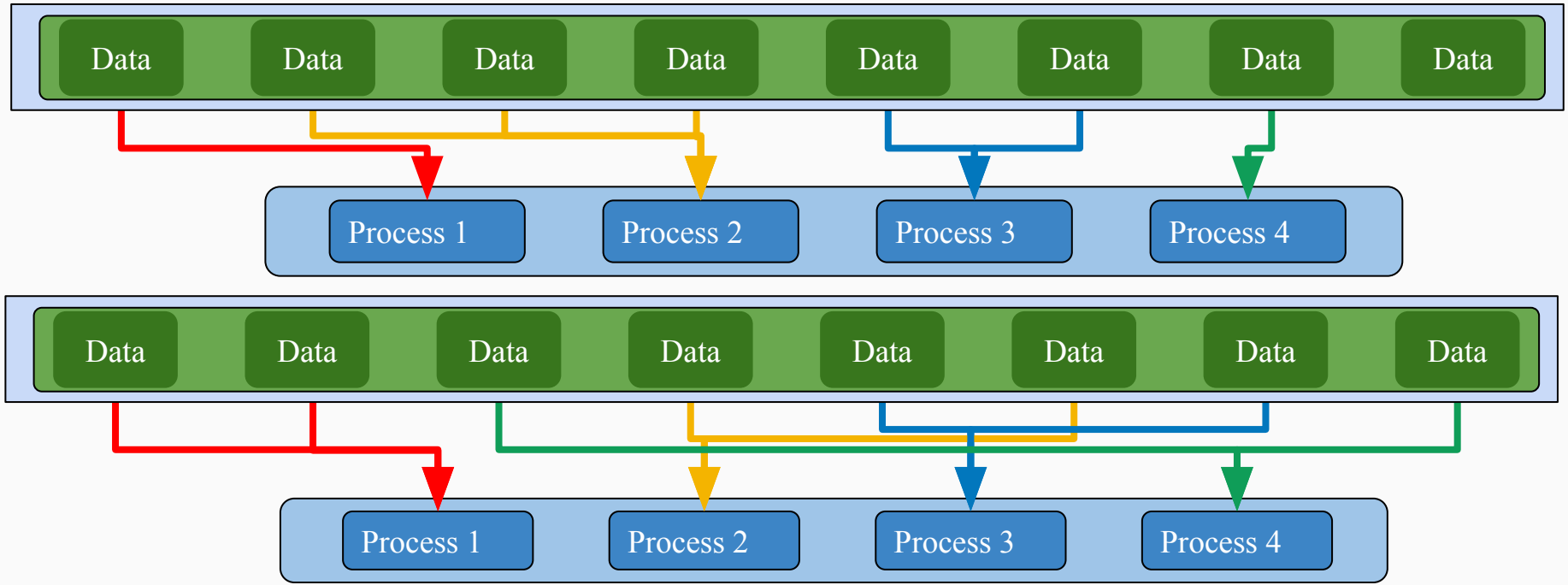
ExSeisDat: A Seismic Parallel I/O Library for Increasing Developer Productivity

Parallel I/O Challenges for Seismic

- Mixture of I/O patterns. e.g
  - Consecutive access of traces
  - Non-monotonic access, non-contiguous, non-consecutive access
- MPI usage common in O&G codes: MPI and MPI-IO limits → ~2 GiB per call
- Collective I/O → balancing of calls on each process.
  - Mismatch → deadlock

Parallel I/O Challenges for Seismic (contd)



Parallel I/O Challenges for Seismic (contd)

- Conformance to the SEG-Y standard is variable.
- Trace data may be stored in obsolete IBM floating point format.
- Difficult to have all of readability, maintainability and scaling/performance of seismic processing codes without substantial effort!

Parallel I/O Challenges for Seismic (contd)

At Tullow Oil PLC:

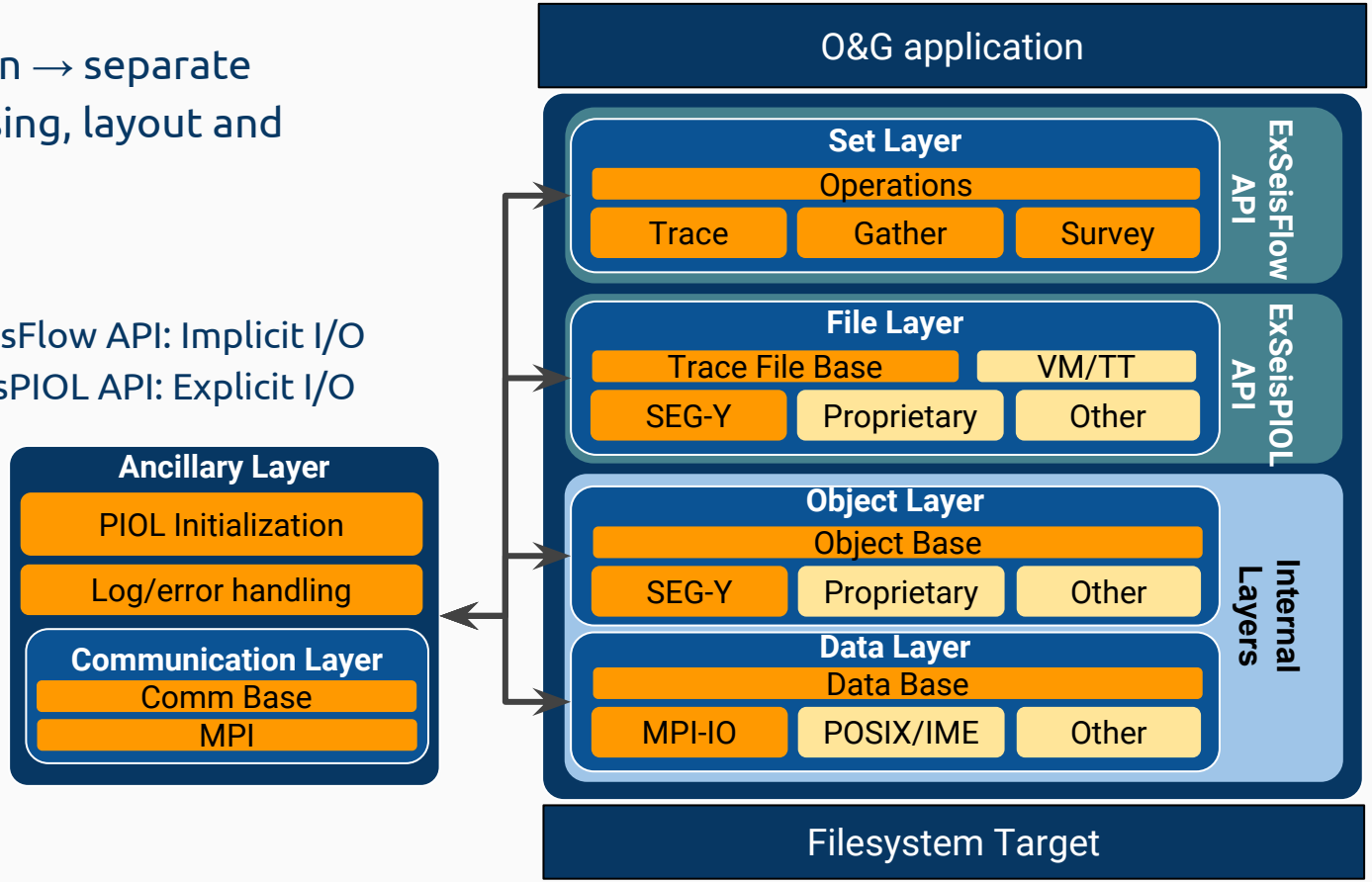
- I/O accounts for 35-50% of total source code
- Effort on I/O source files is 50% less than non-I/O source files
- I/O related commits equal to non-I/O related commits

ExSeisDat

- **Ex**treme-Scale **Seis**mic **Dat**a Library (**ExSeisDat**)
- Easy to use, geophysicist-friendly C++ and C APIs with planned Python support
- Scalable / Performance
- Reduces maintenance → Reduces codebase sizes substantially
- Extensive testing framework (unit, integration and system tests)

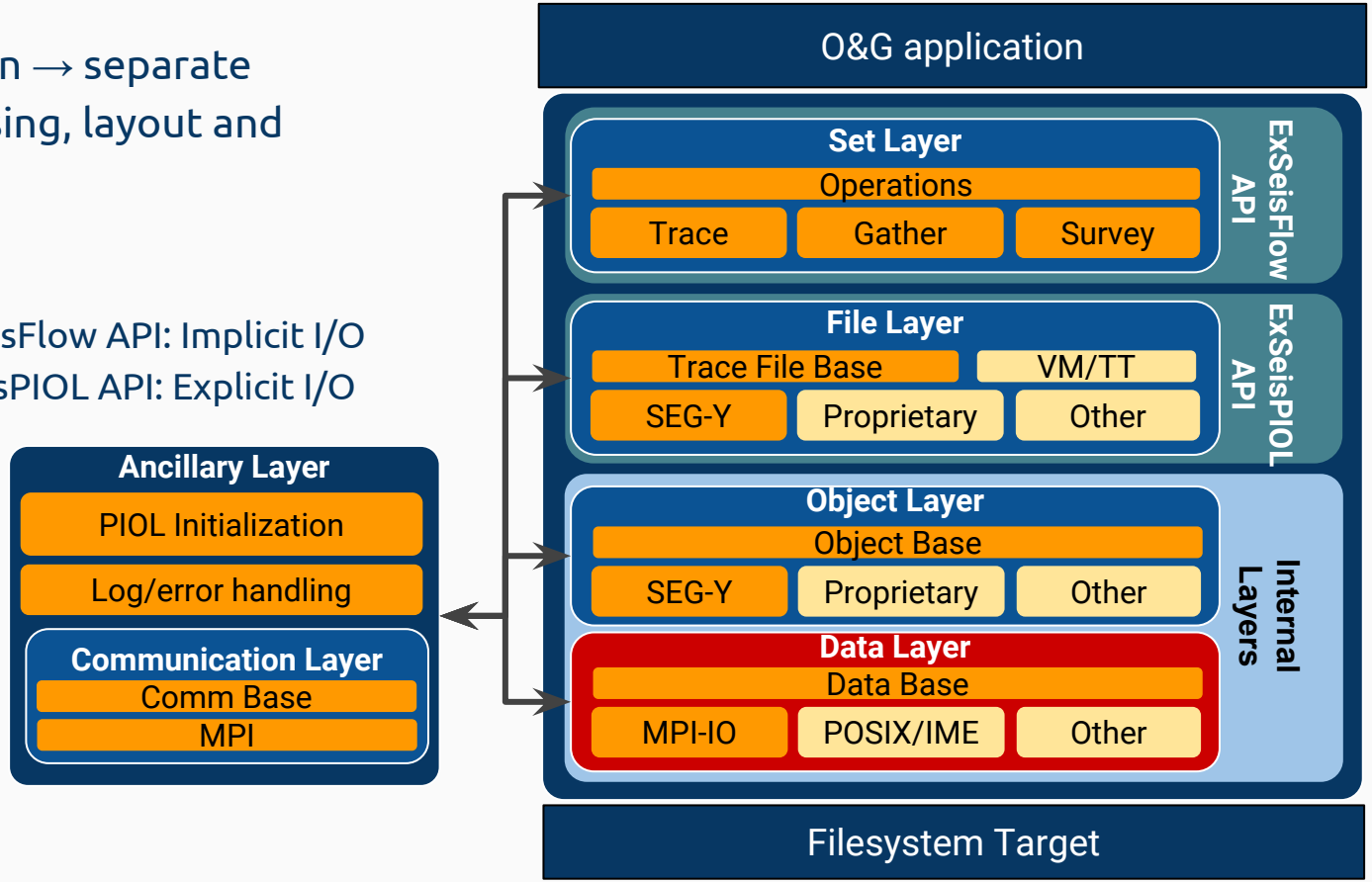
ExSeisPIOL Design

- Multi-Layer solution → separate file-format processing, layout and MPI-IO details.
- Two public APIs:
  - High-level ExSeisFlow API: Implicit I/O
  - Low-level ExSeisPIOL API: Explicit I/O
- Computational geophysicists / software engineers writing seismic processing software on HPC clusters



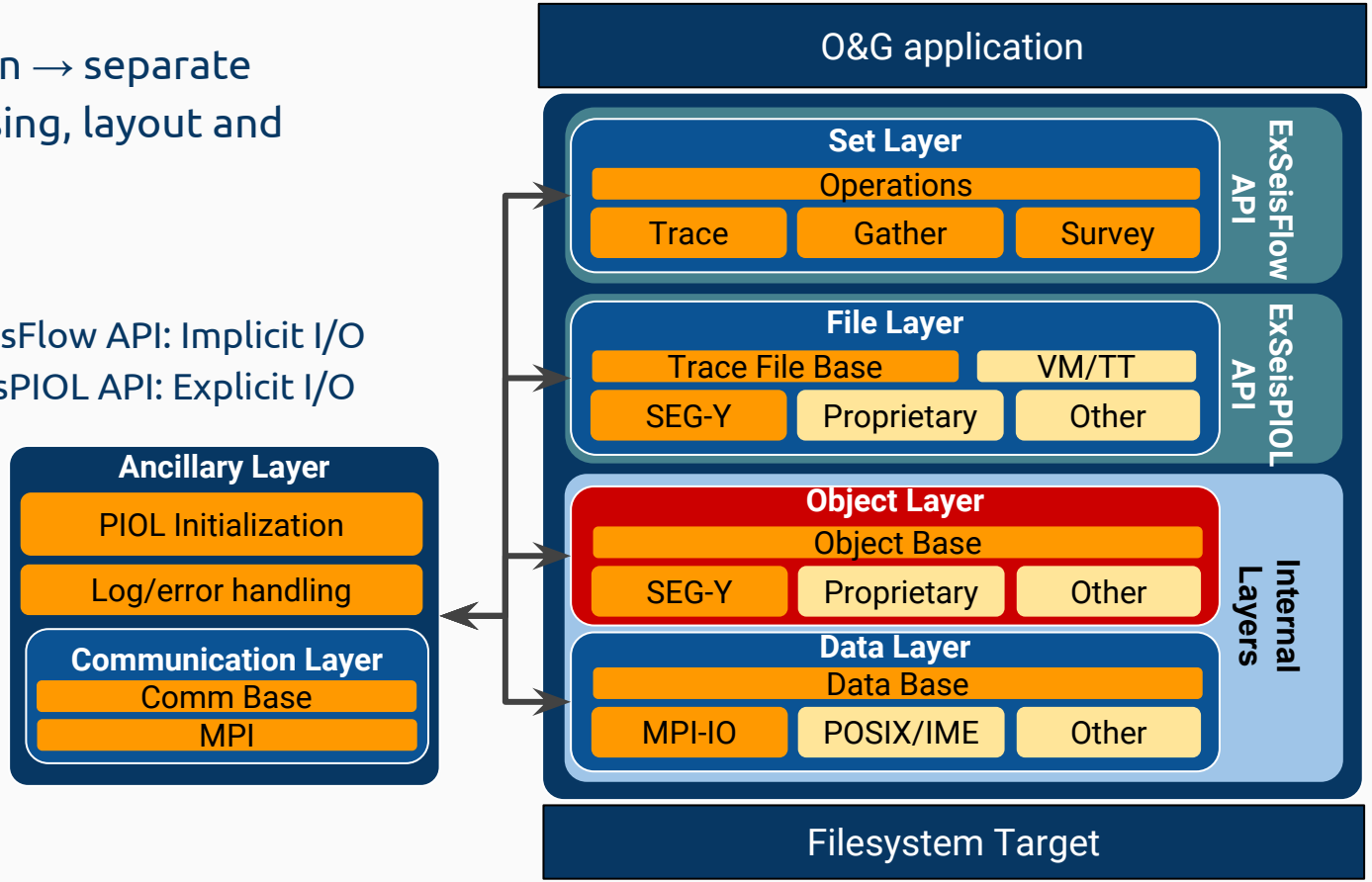
ExSeisPIOL Design

- Multi-Layer solution → separate file-format processing, layout and MPI-IO details.
- Two public APIs:
  - High-level ExSeisFlow API: Implicit I/O
  - Low-level ExSeisPIOL API: Explicit I/O
- Computational geophysicists / software engineers writing seismic processing software on HPC clusters



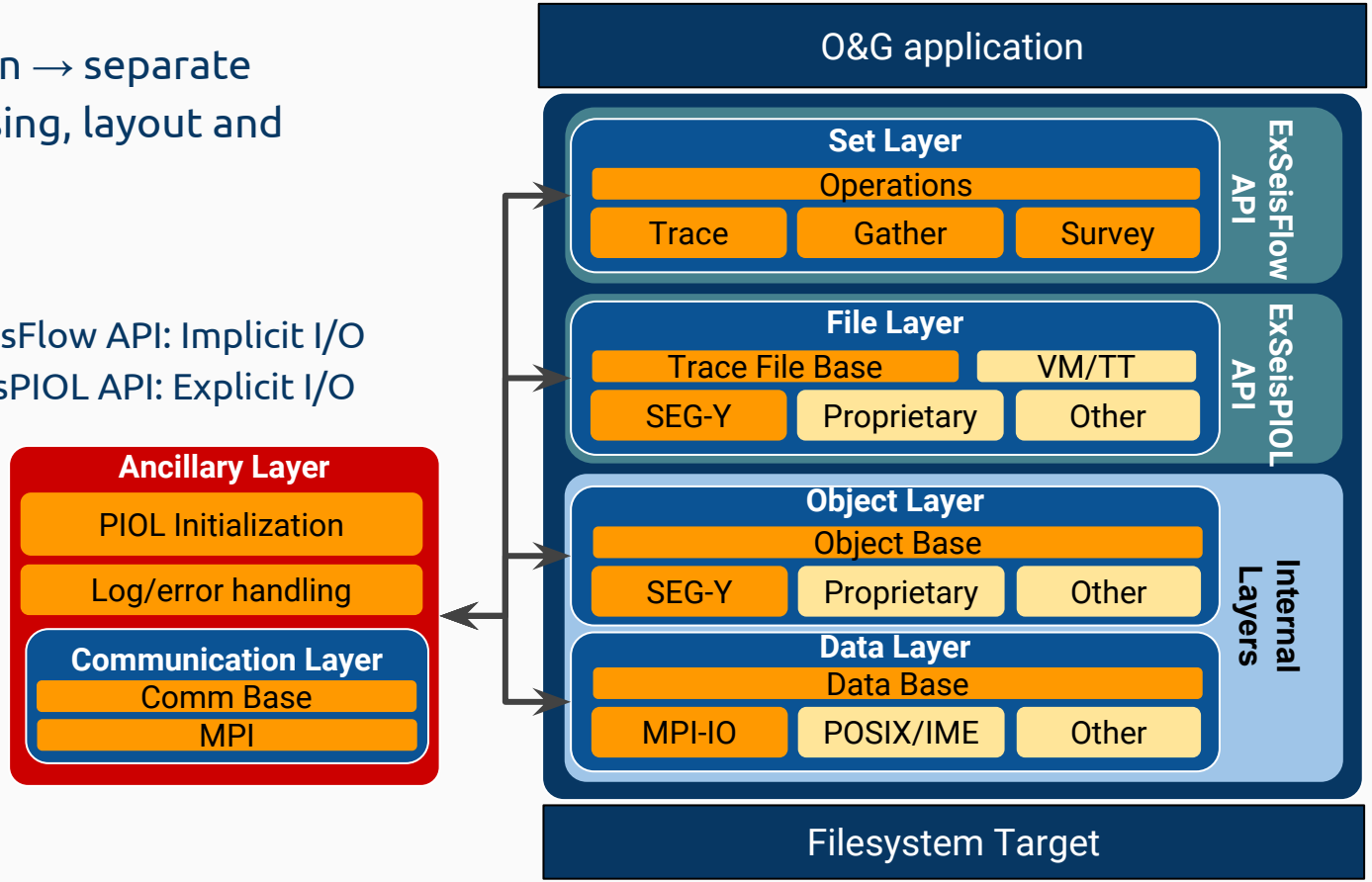
ExSeisPIOL Design

- Multi-Layer solution → separate file-format processing, layout and MPI-IO details.
- Two public APIs:
  - High-level ExSeisFlow API: Implicit I/O
  - Low-level ExSeisPIOL API: Explicit I/O
- Computational geophysicists / software engineers writing seismic processing software on HPC clusters



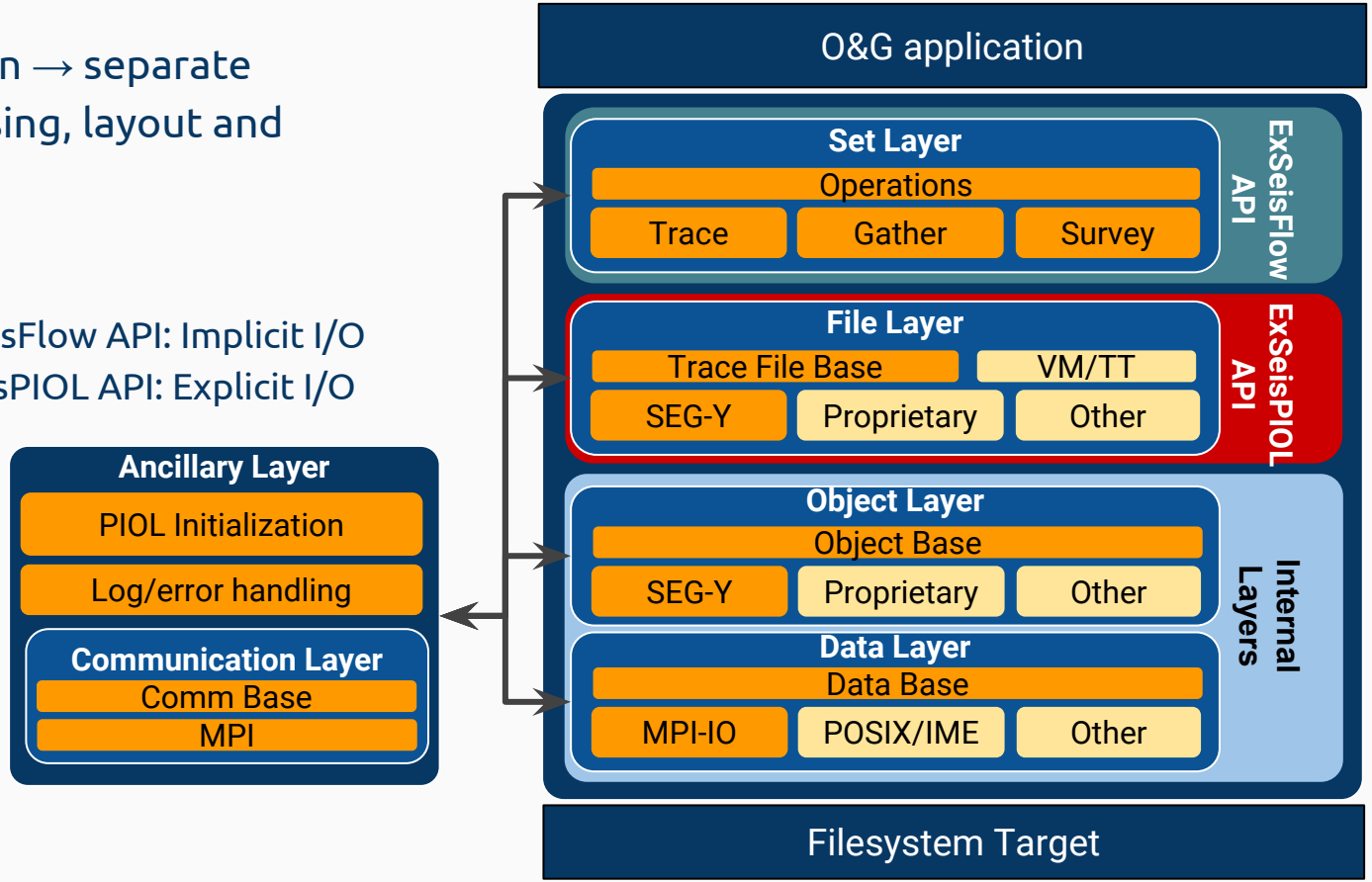
ExSeisPIOL Design

- Multi-Layer solution → separate file-format processing, layout and MPI-IO details.
- Two public APIs:
  - High-level ExSeisFlow API: Implicit I/O
  - Low-level ExSeisPIOL API: Explicit I/O
- Computational geophysicists / software engineers writing seismic processing software on HPC clusters



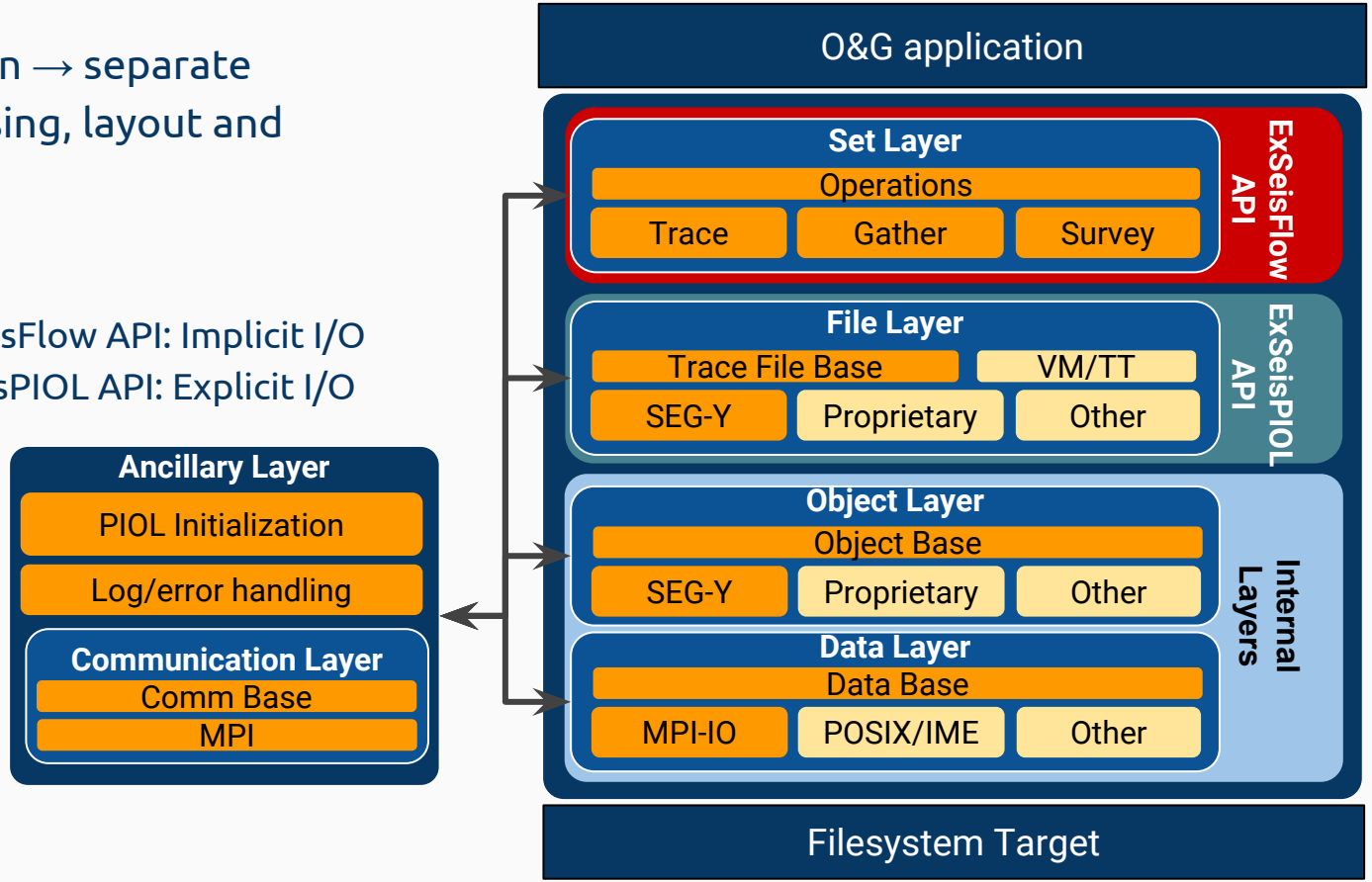
ExSeisPIOL Design

- Multi-Layer solution → separate file-format processing, layout and MPI-IO details.
- Two public APIs:
  - High-level ExSeisFlow API: Implicit I/O
  - Low-level ExSeisPIOL API: Explicit I/O
- Computational geophysicists / software engineers writing seismic processing software on HPC clusters



ExSeisPIOL Design

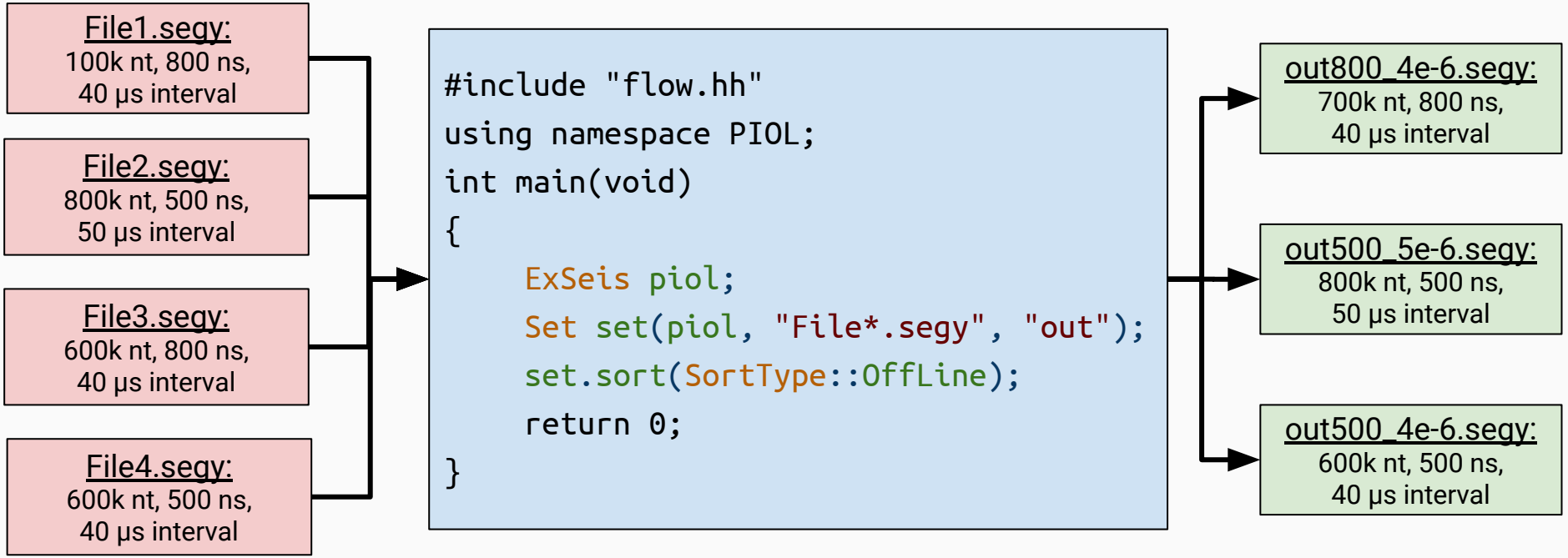
- Multi-Layer solution → separate file-format processing, layout and MPI-IO details.
- Two public APIs:
  - High-level ExSeisFlow API: Implicit I/O
  - Low-level ExSeisPIOL API: Explicit I/O
- Computational geophysicists / software engineers writing seismic processing software on HPC clusters



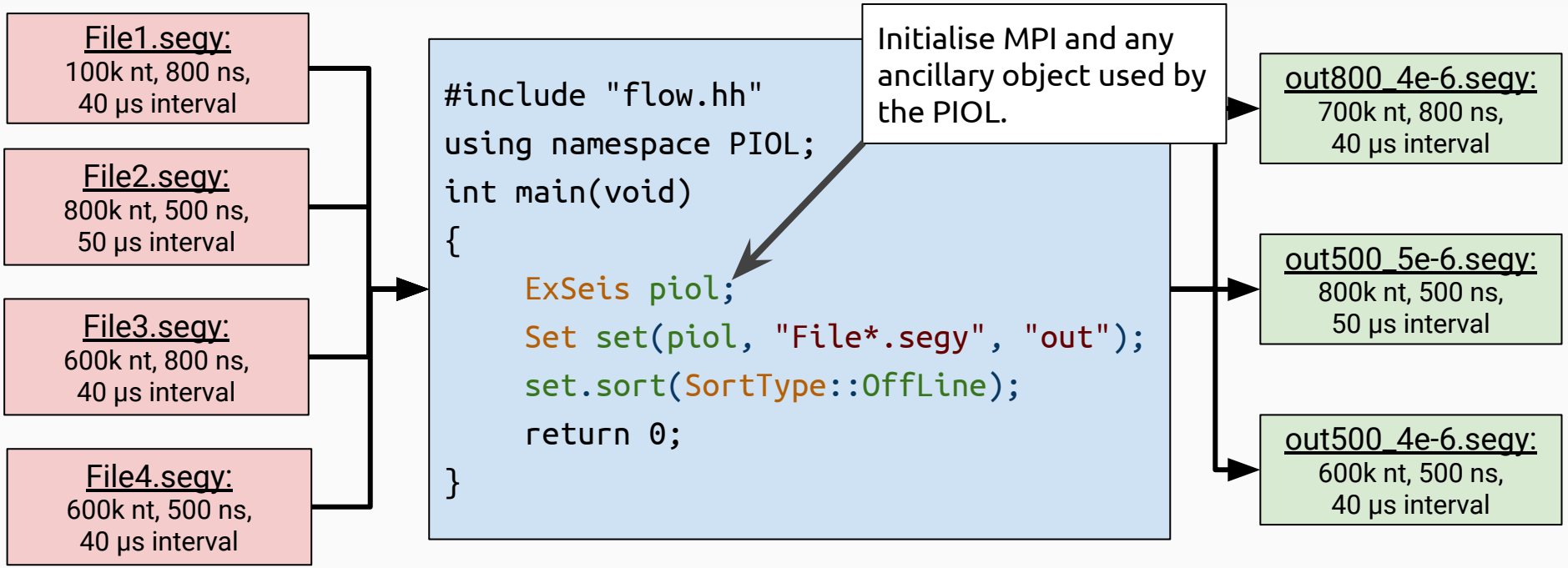
Available ExSeisFlow Operations Include

- File Concatenation
- Sorting
- 4D Binning
- Trace Muting and Filtering
- Trace Transforms including Radon to Angle
- Automatic Gain Control

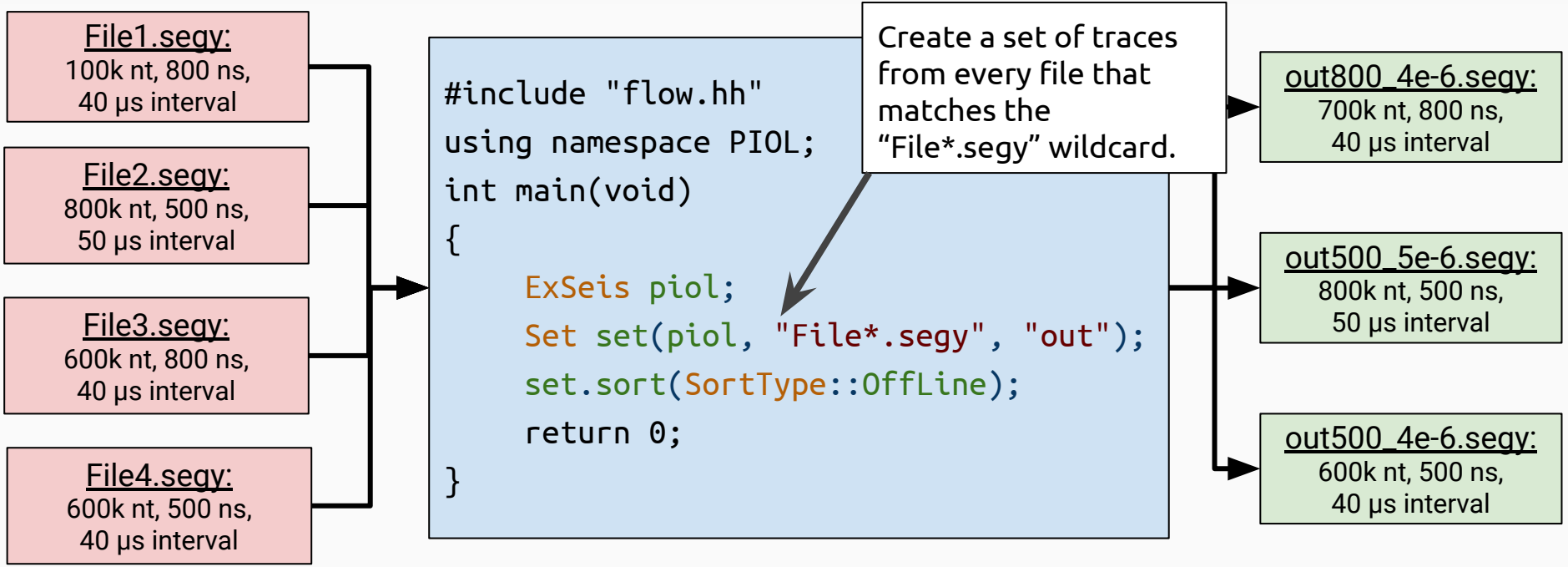
ExSeisFlow Example: Concatenation & Sorting (C++)



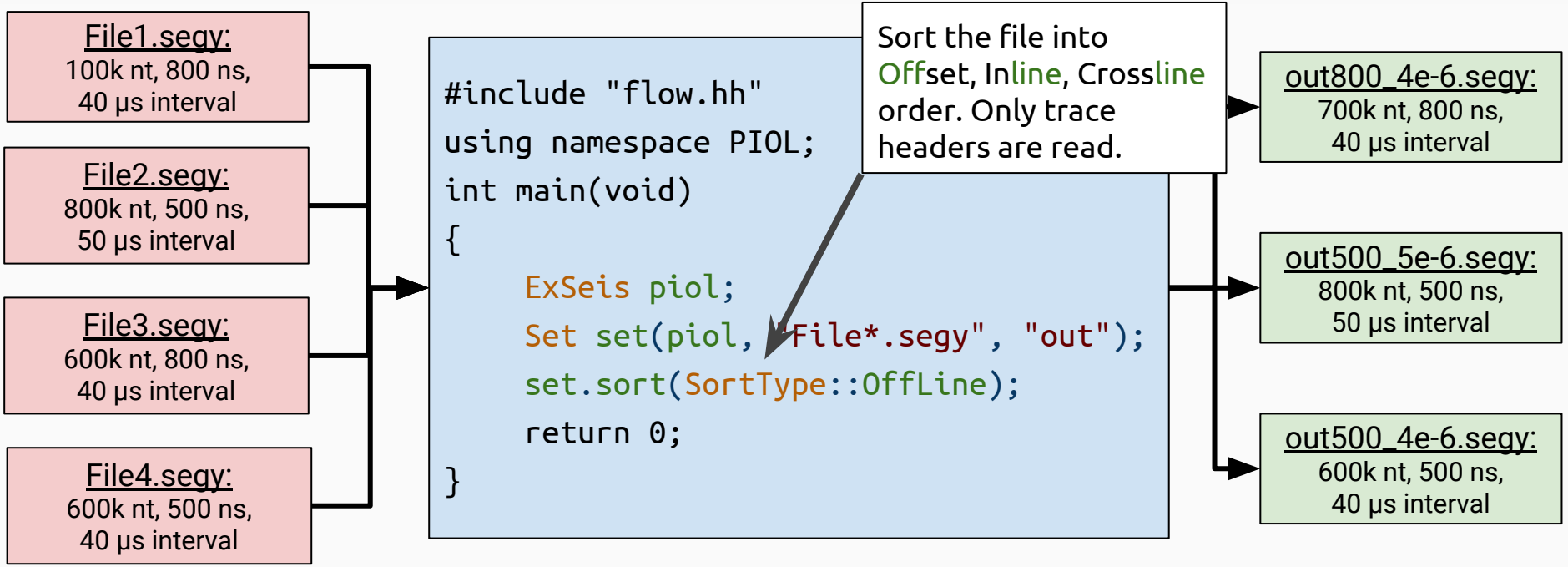
ExSeisFlow Example: Concatenation & Sorting (C++)



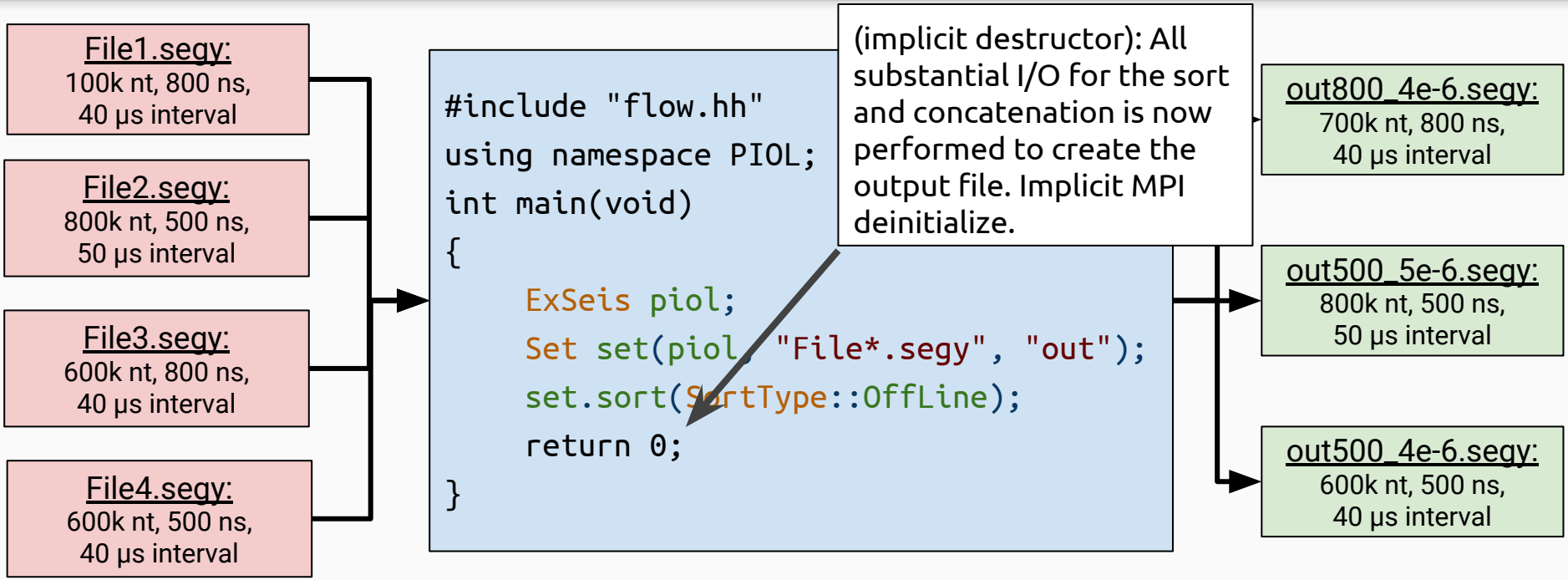
ExSeisFlow Example: Concatenation & Sorting (C++)



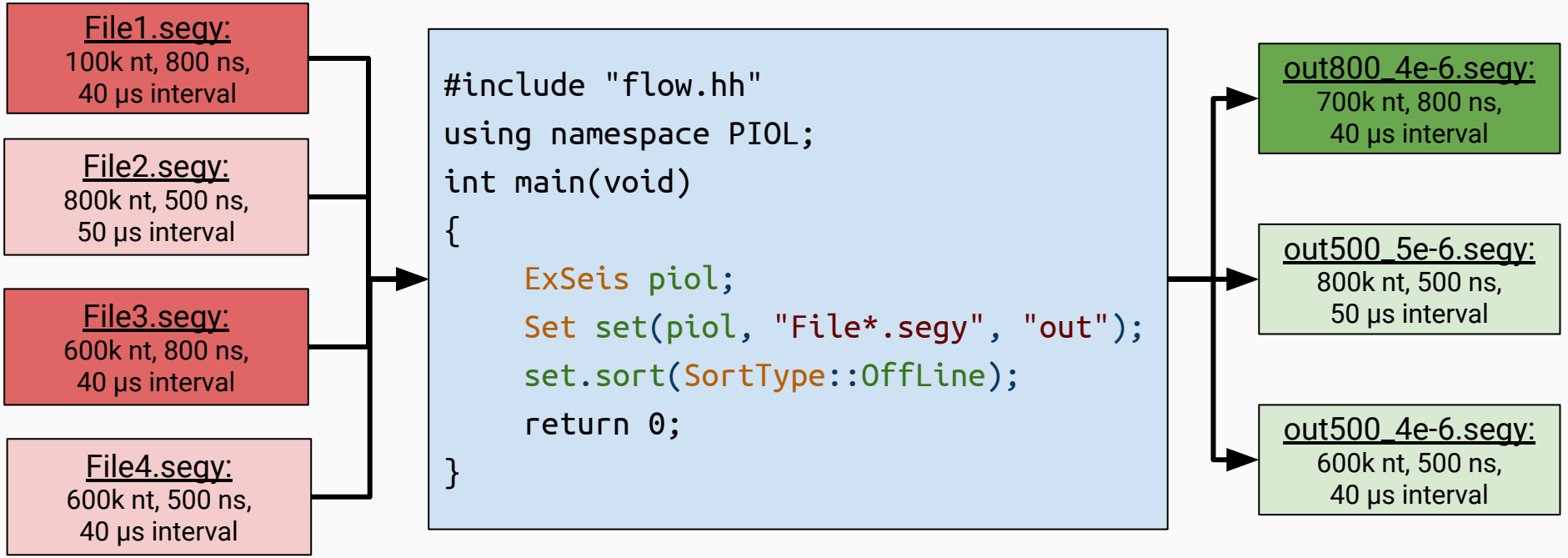
ExSeisFlow Example: Concatenation & Sorting (C++)



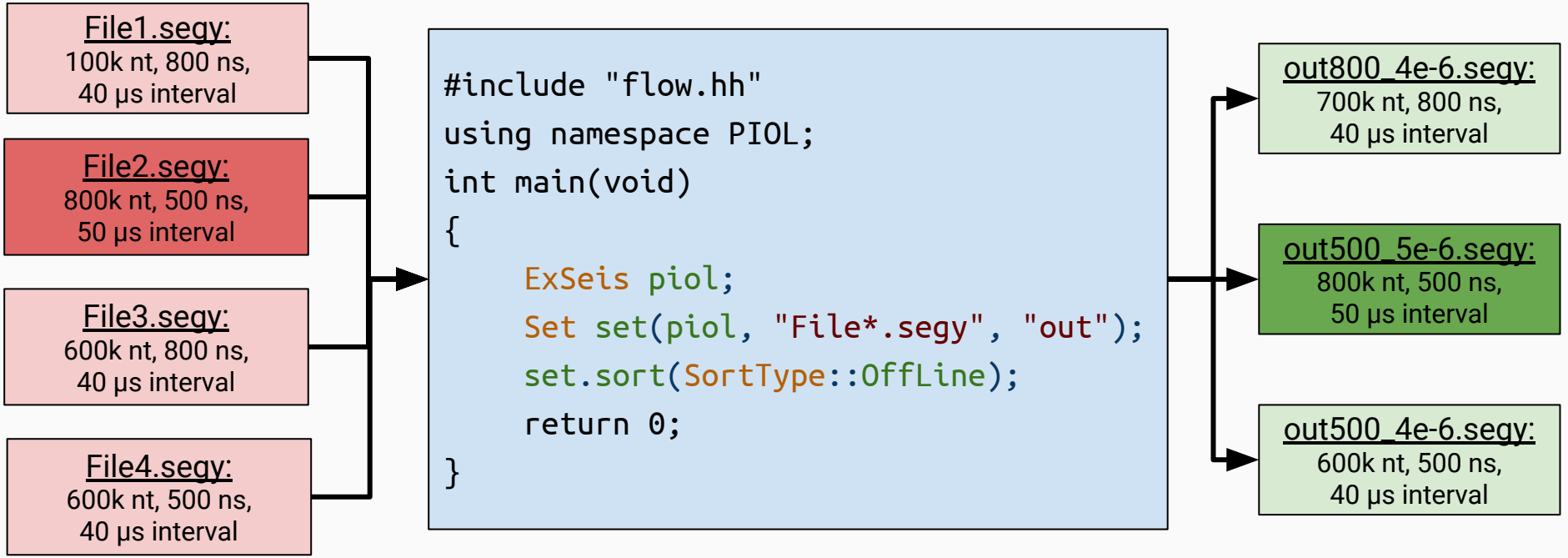
ExSeisFlow Example: Concatenation & Sorting (C++)



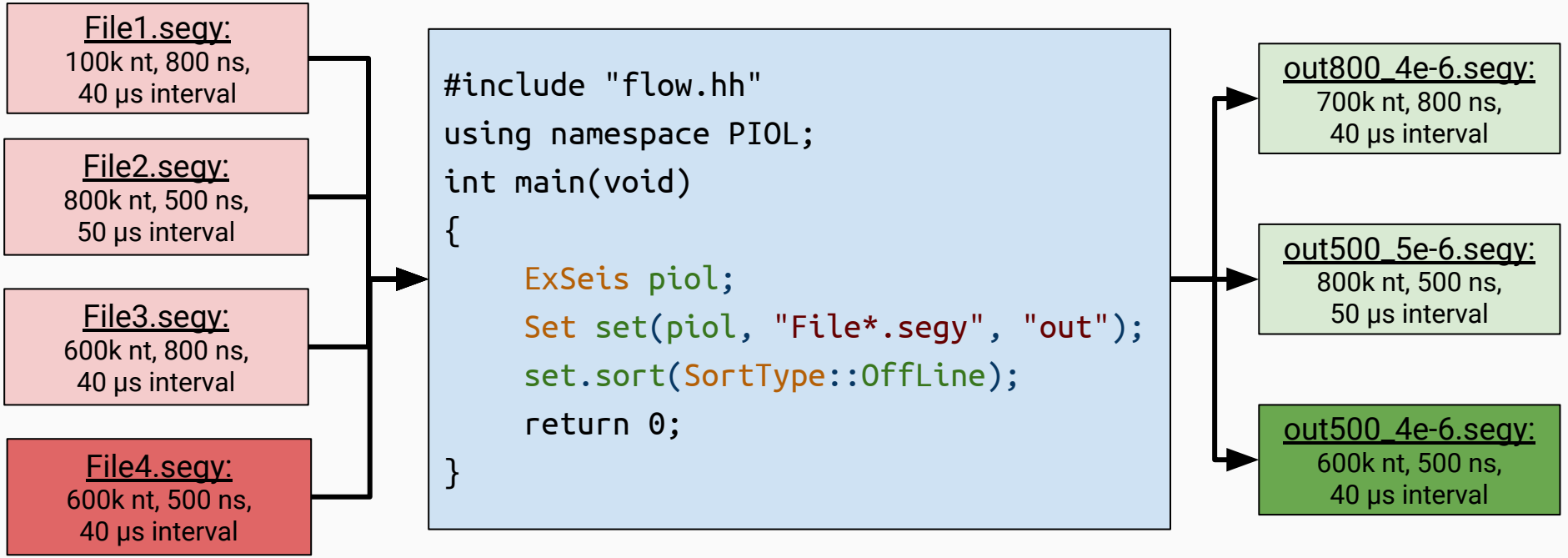
ExSeisFlow Example: Concatenation & Sorting (C++)



ExSeisFlow Example: Concatenation & Sorting (C++)



ExSeisFlow Example: Concatenation & Sorting (C++)



Integrating ExSeisDat & Existing Applications

- Ideal for applications with heavy or complicated I/O
- Example: Kirchhoff Migration (KTMig)
  - **Performance Tests KTMig**
    - System test: 183 s -> 182 seconds
    - Optim Test: 737 s -> 791s with no (very very small) write, more read heavy, 120GB file input file, 4x24 procs
  - **Total lines of code**:
    - Total code base: 6304 -> 5326 ~16% reduction
    - mpiUtils.cc, i.e. code related to MPI & I/O: 2749 -> 2051 ~25% reduction

Example: Read Trace Data

Old Version

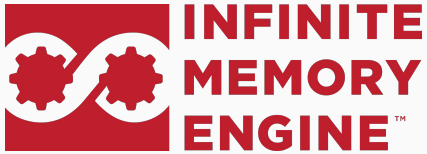
```
MPI_File file = typeDesc_[type].file_;
MPI_Offset disp = getTraceDataOffset( startTrace, ns );

// set the view
MPI_File_set_view( file, disp, MPI_FLOAT, inputFileViews_[round], "native",
MPI_INFO_NULL );
// read the file
llong nt = tracesPerViews_[round];

profile_mpi_file_read( file, &data[0][0], nt * ns, MPI_FLOAT,
MPI_STATUS_IGNORE );

// transform the data according to the FP format
if ( fpFormat == 1 ) { // IBM format
   ibm2ieee( &data[0][0], &data[0][0], nt * ns );
}
else if ( fpFormat == 5 ) { // IEEE Big endian format
   byteSwapData( &data[0][0], nt * ns );
}
else { // we don't know how to deal with that
   if ( myRank_ == 0 ) {
      cout << "Error in reading the input SEGY survey file:\n"
         << " FP format " << fpFormat << " is not supported\n"
         << " Aborting\n";
   }
   MPI_Finalize();
   exit( 1 );
}
```

New Version

```
piolInFile_[INPUT]->readTrace( tracesPerViews_[round],
        reinterpret_cast<size_t *>( &localIdxs_[tracesPerPrevViews_[round]] ), data[0] );
```

DDN Storage IME: Burst-Buffer Technology

- ExSeisDat provides hardware specific optimisations
  - So geophysicists don’t have to!
- DDN IME is a next-generation tiered data-storage architecture
- High-throughput SSDs
- Smart IME Software:
  - Non-contiguous write performance ≈ contiguous write performance



Burst Buffer: I/O Workflow

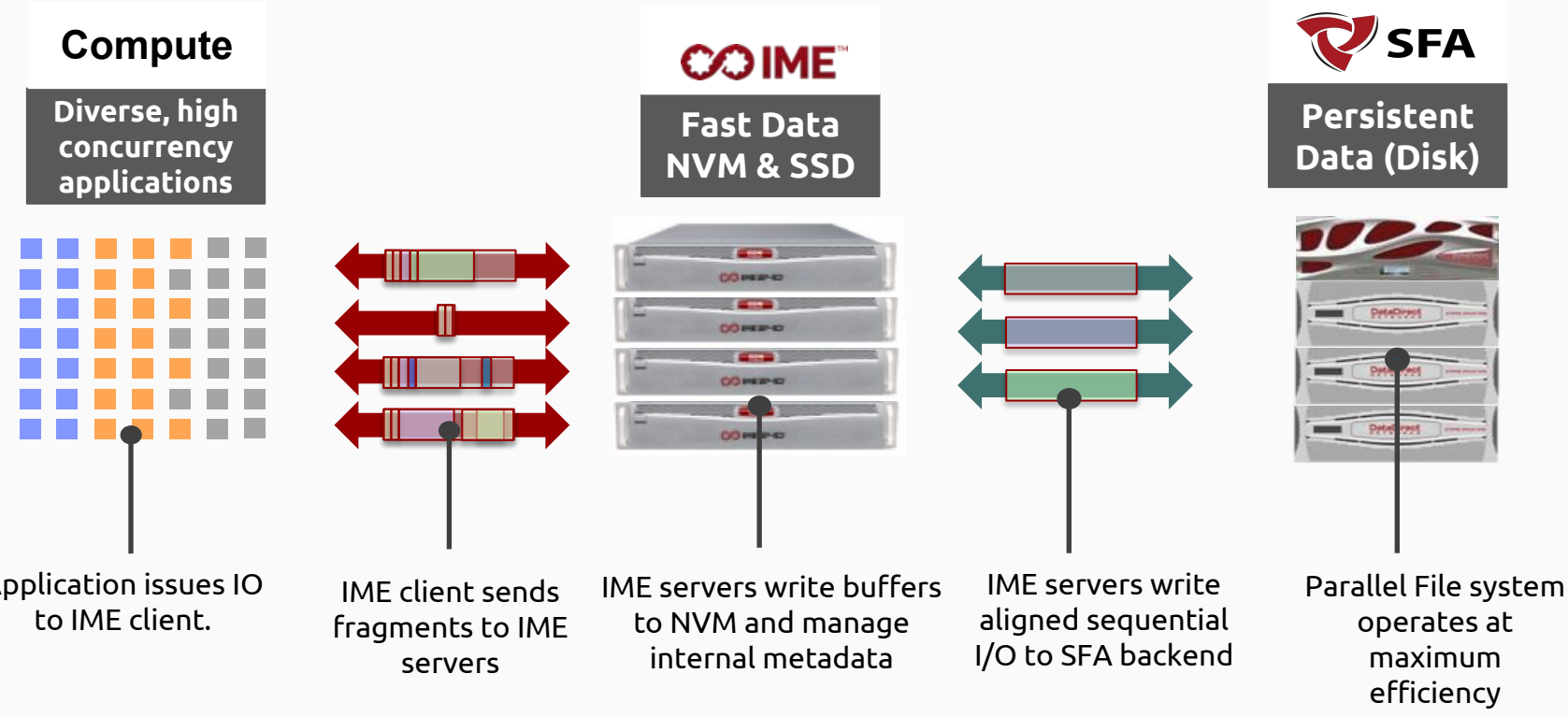
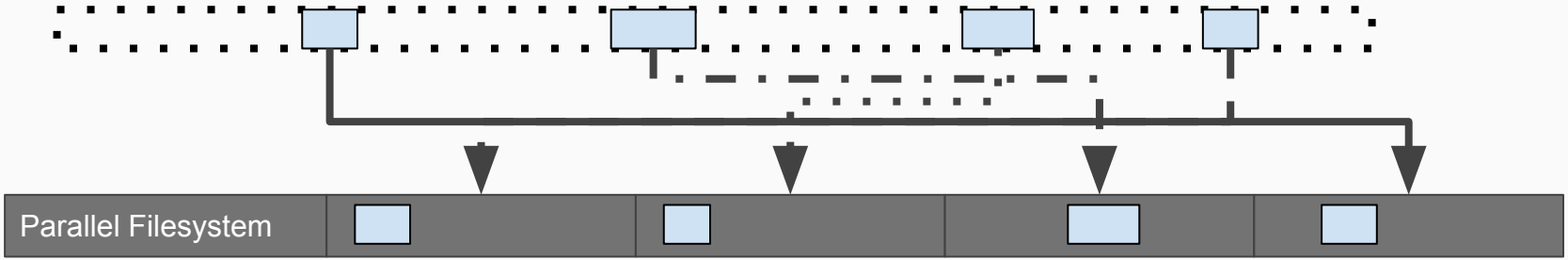
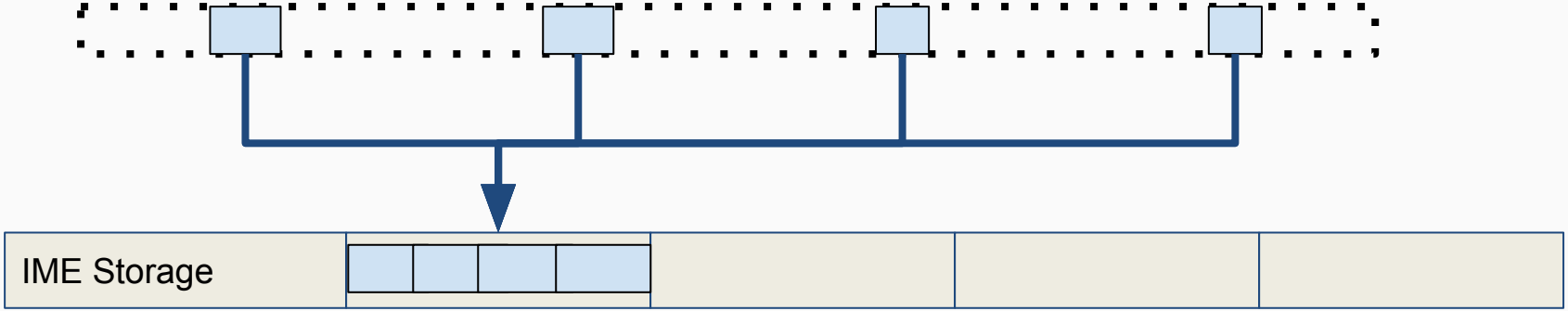


Image courtesy of DDN Storage.

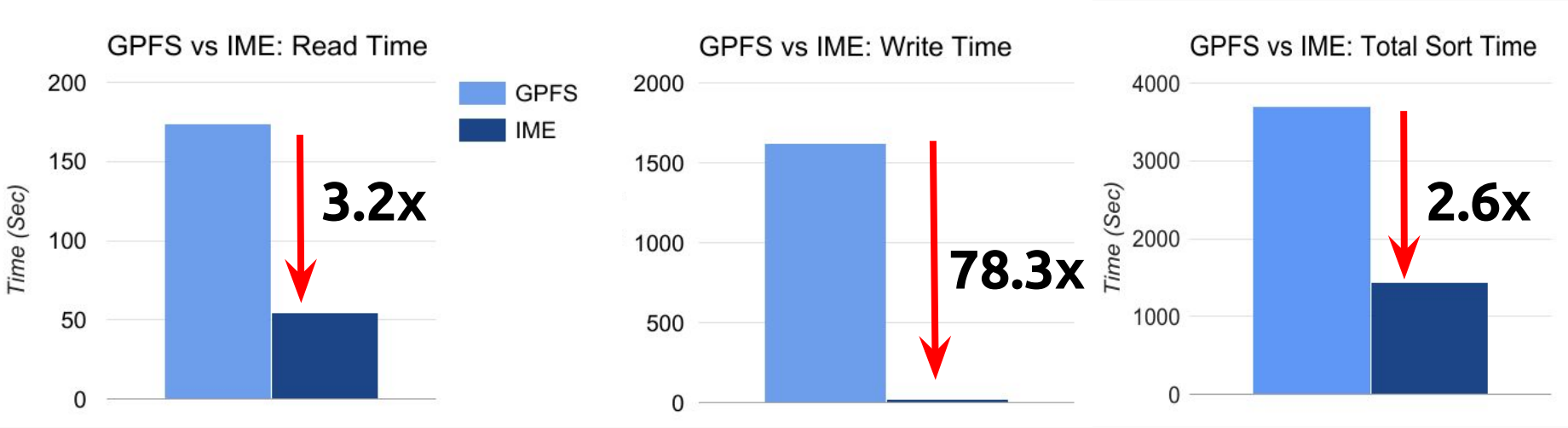
Treatment of I/O patterns, Parallel File-system vs Burst Buffer



IME: Contention free and less redundant reads



SEG-Y Sort Benchmarking - DDN Test Platform



- Comparing GPFS against an IME setup.
- Sort of a 400 GiB SEG-Y file (400 GiB read, 400 GiB write), 4 nodes.
- Total Time: 63% GPFS I/O, only 5% IME I/O for sort
- DDN IME is more than a SSD bank! **Software advantage** over GPFS

Conclusions

- Parallel I/O Library for seismology workflows
- SEG-Y compatible
- ExSeisPIOL and ExSeisFlow can use DDN IME hardware for big speedups
- Easy to use API → increases productivity
- Production ready but development ongoing

Accessing the ExSeisDat Library

- ExSeisDat is ***currently*** available at

  www.ichec.ie/partnerships/industry/exseisdat

- Open source → LGPL 3.0

Thanks for listening

Cathal Ó Broin [a,b], Ruairi Short [a], Meghan Fisher [a,b], Seán Delaney [c], Steven Dagg [c], Gareth O'Brien [c], Jean-Thomas Acquaviva [d], Michael Lysaght [a,b]

a. ICHEC, Dublin Ireland
b. Lero, Dublin, Ireland
c. Tullow Oil, Dublin, Ireland
d. DDN Storage, Paris, France

Funding: