Enabling Compute HW With oneAPI & UXL-F

Alex Pim (Imagination)

UXL
Unified Acceleration Foundation

Compute Product Philosophy

The Customer Development Journey

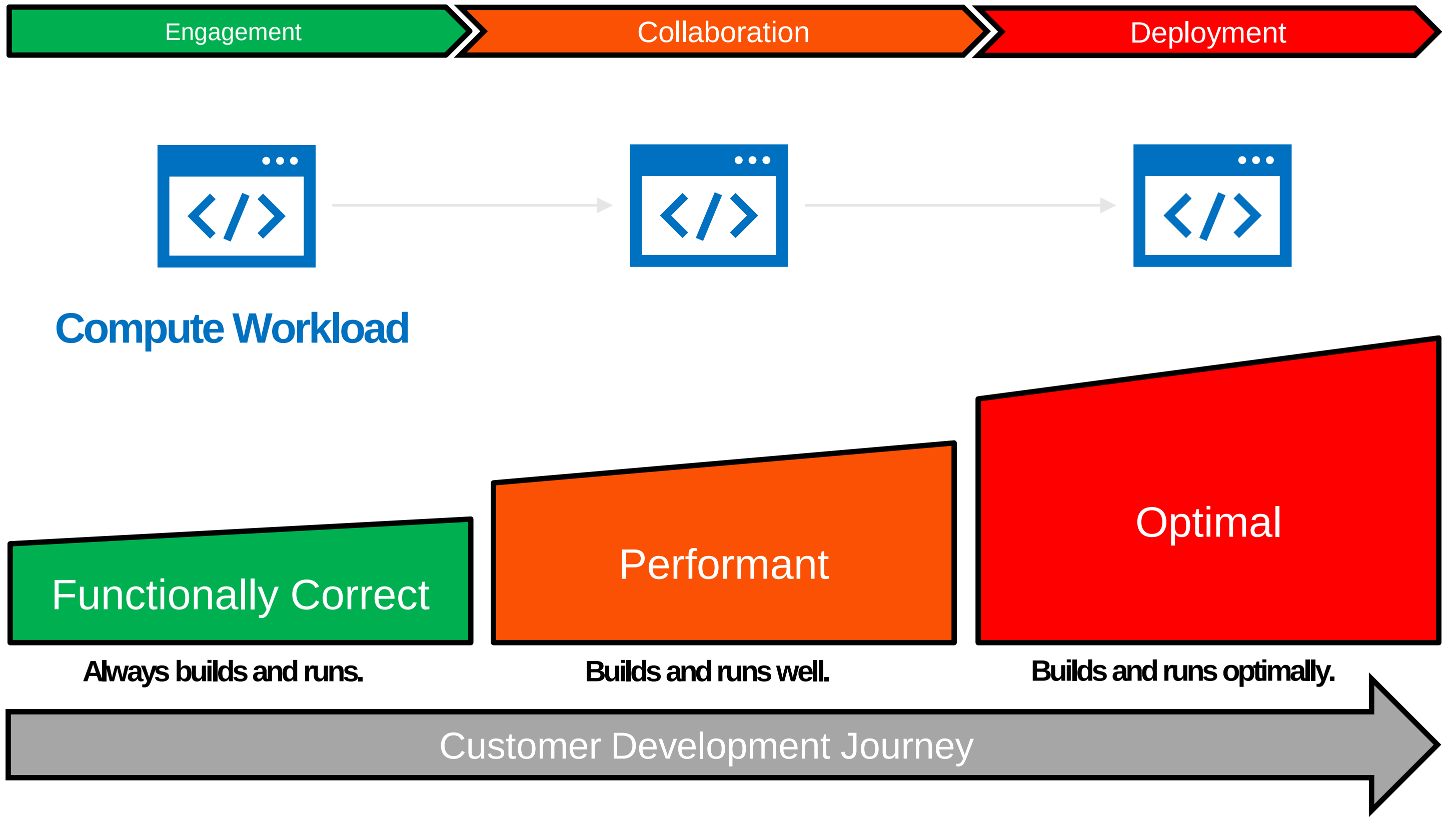
Analysis of:

- *Customer engagements,*
- *Customer feedback and, the*
- *Competitive landscape for compute*

Suggests that customers need a clear development journey where their workloads move from:

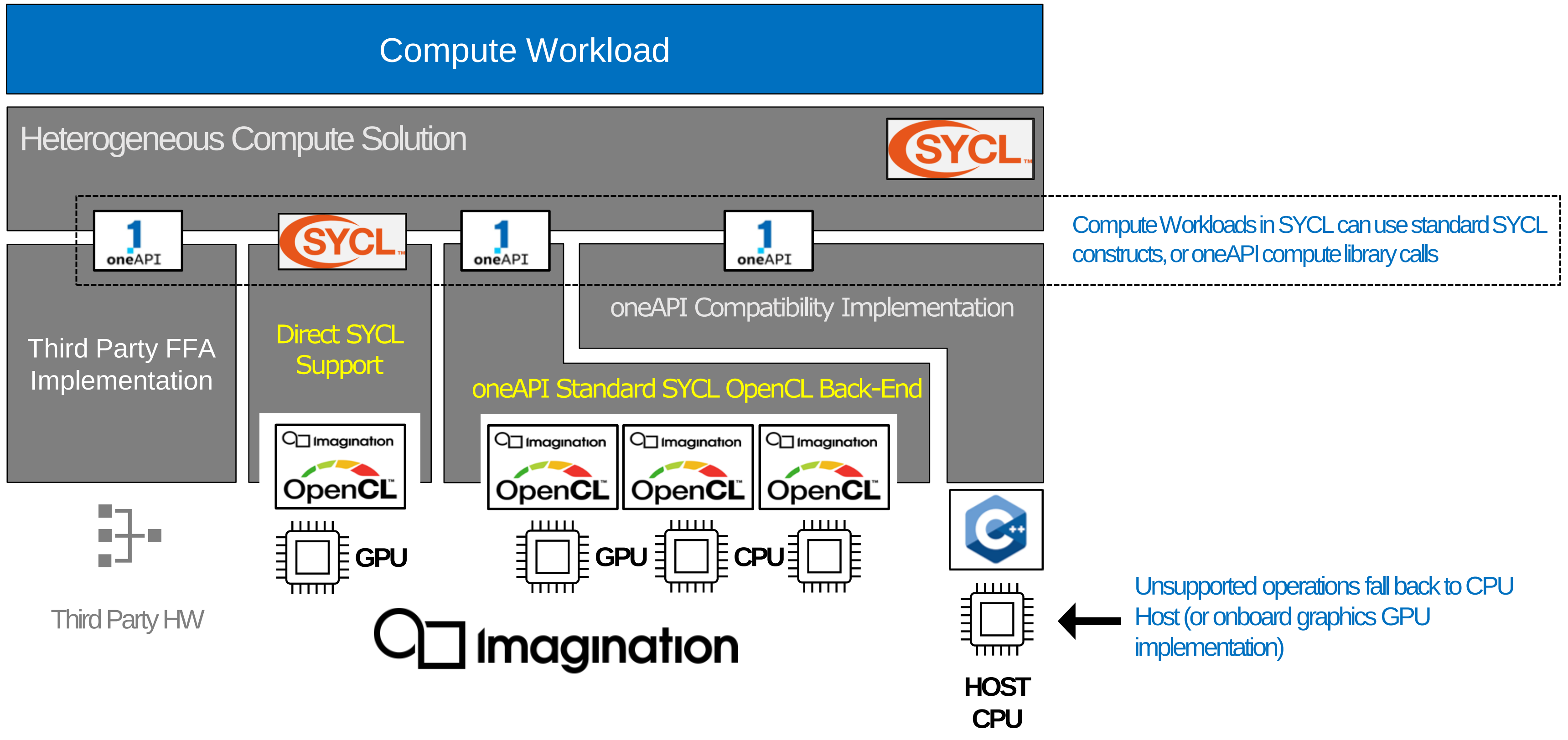
- *Supported & Functionally correct (during engagement), to*
- *Performant (during collaboration), and*
- *Optimal (during deployment)*

Therefore, a clear, efficient and well documented workflow for customers is needed → <u>and oneAPI can facilitate this.</u>

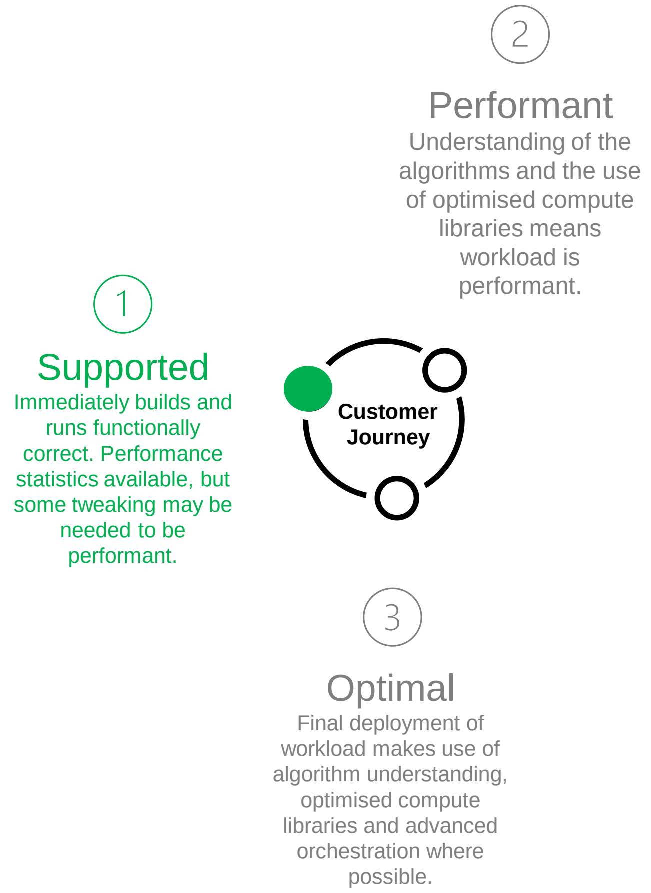


Hardware Acceleration using oneAPI

An example of a supported compute workload using the standard SYCL OpenCL Back-end



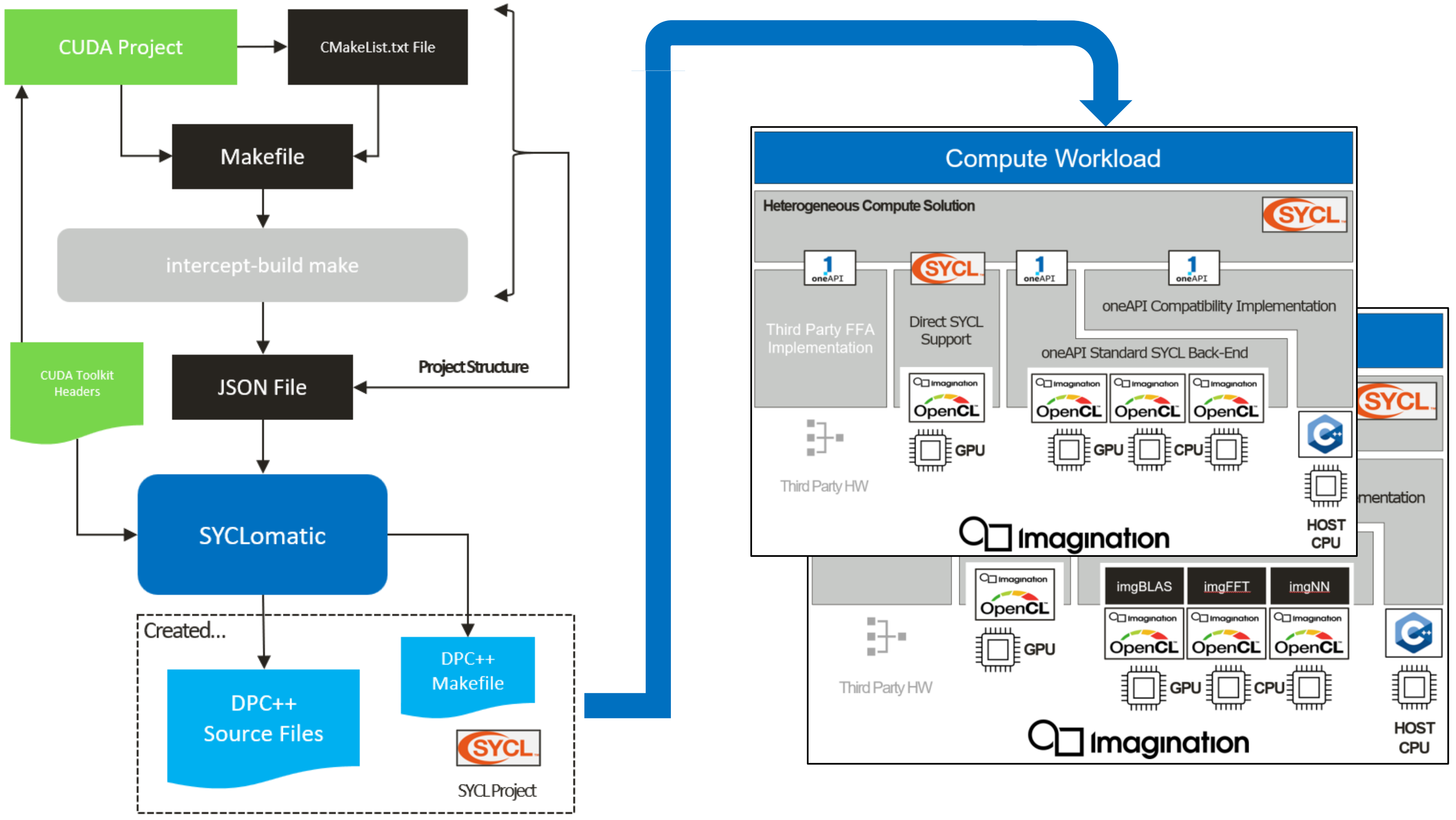
Compute Workload
Heterogeneous Compute Solution
SYCL
oneAPI
SYCL
oneAPI
oneAPI
Compute Workloads in SYCL can use standard SYCL constructs, or oneAPI compute library calls
oneAPI Compatibility Implementation
Third Party FFA Implementation
Direct SYCL Support
oneAPI Standard SYCL OpenCL Back-End
Imagination OpenCL
Imagination OpenCL
Imagination OpenCL
Imagination OpenCL
GPU
GPU
CPU
Third Party HW
Imagination
HOST CPU
Unsupported operations fall back to CPU Host (or onboard graphics GPU implementation)

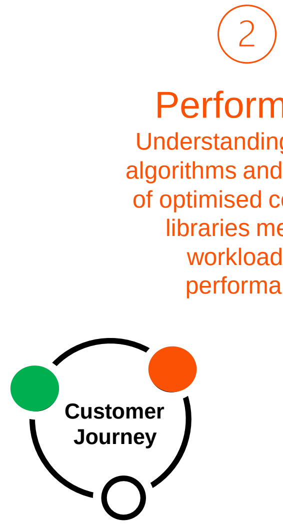


2
Performant
Understanding of the algorithms and the use of optimised compute libraries means workload is performant.
1
Supported
Immediately builds and runs functionally correct. Performance statistics available, but some tweaking may be needed to be performant.
Customer Journey
3
Optimal
Final deployment of workload makes use of algorithm understanding, optimised compute libraries and advanced orchestration where possible.

Hardware Acceleration using oneAPI

An example of a supported *and* performant workload via CUDA conversion using SYCLomatic and oneAPI





2

Performant
Understanding of the algorithms and the use of optimised compute libraries means workload is performant.

1

Supported
Immediately builds and runs functionally correct. Performance statistics available, but some tweaking may be needed to be performant.

3

Optimal
Final deployment of workload makes use of algorithm understanding, optimised compute libraries and advanced orchestration where possible.

The oneAPI vendor back-end interface

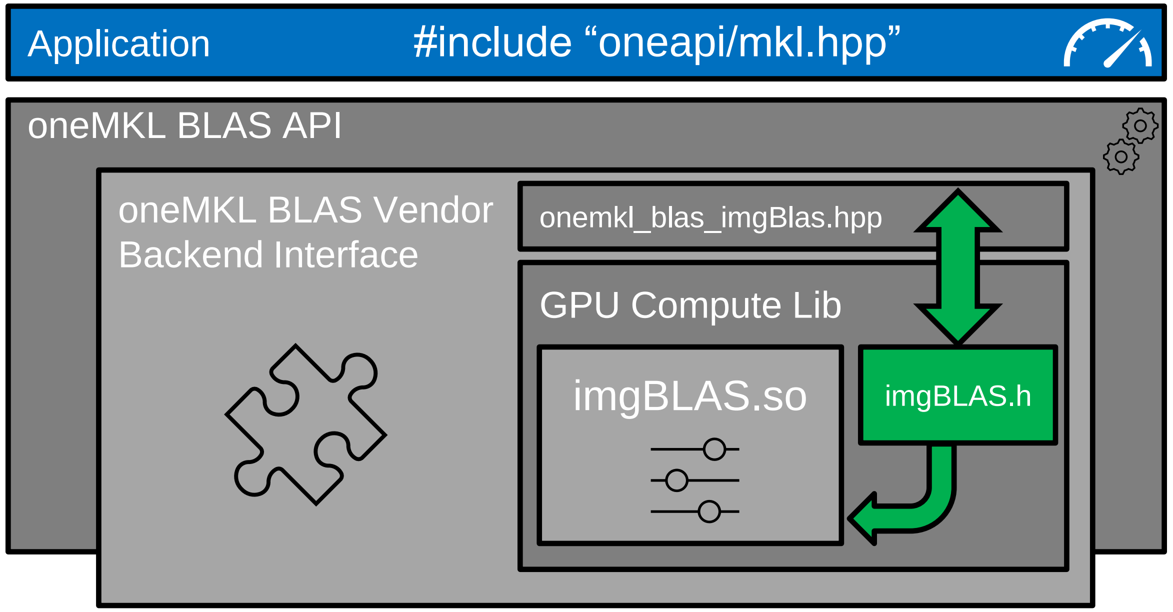
Easy Third-Party Hardware IP Integration

This allows:

- Optimal implementations of the operations you want to accelerate on your hardware for customers
- The provision of a fallback to CPU functionality to ensure build and execution continuity across all hardware platforms
- Optimised hardware functions for your IP can be in a different language to SYCL (e.g., OpenCL)

oneMKL Vendor Backend:

- Clear, structured and intuitive layering of API's from oneMKL (e.g., BLAS) through to your own BLAS implementation in SYCL or OpenCL.



oneAPI CUDA library equivalents

An example of a library-to-library API call mapping

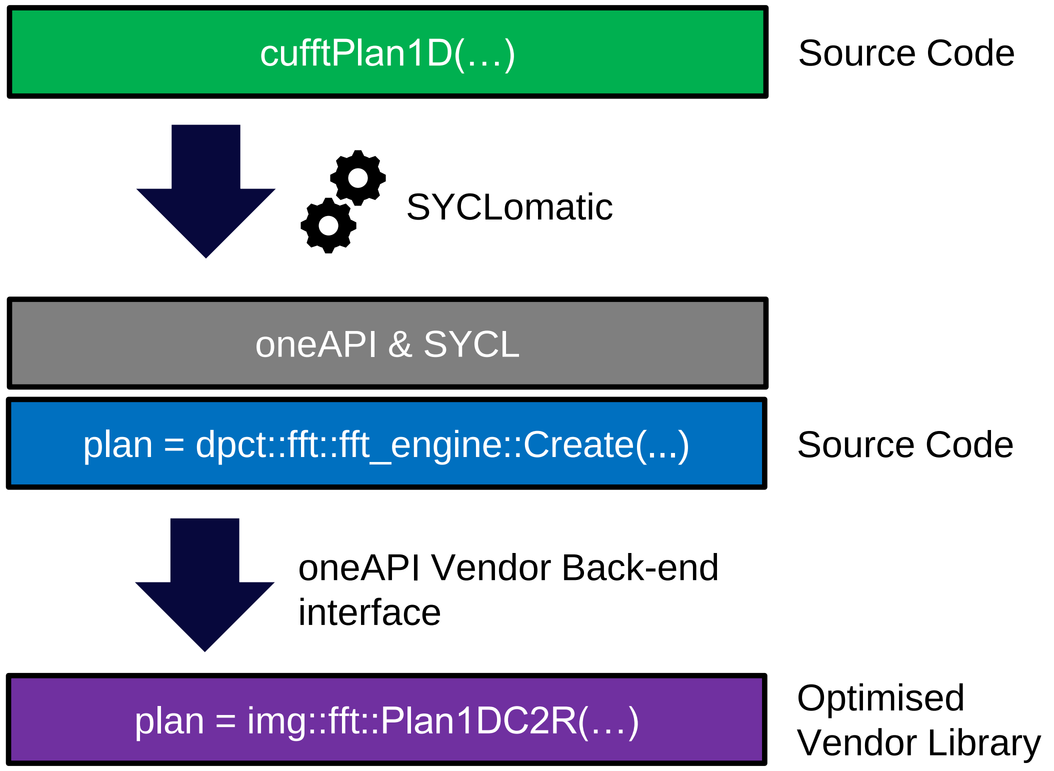
oneMKL provides equivalent compute libraries to Nvidia software library components:

- *cuBLAS → oneMKL BLAS*
- ***cuFFT → oneMKL FFT***
- *cuDNN → oneDNN*

…which GPU providers can then support using compute libraries and the oneAPI vendor back-end interface:

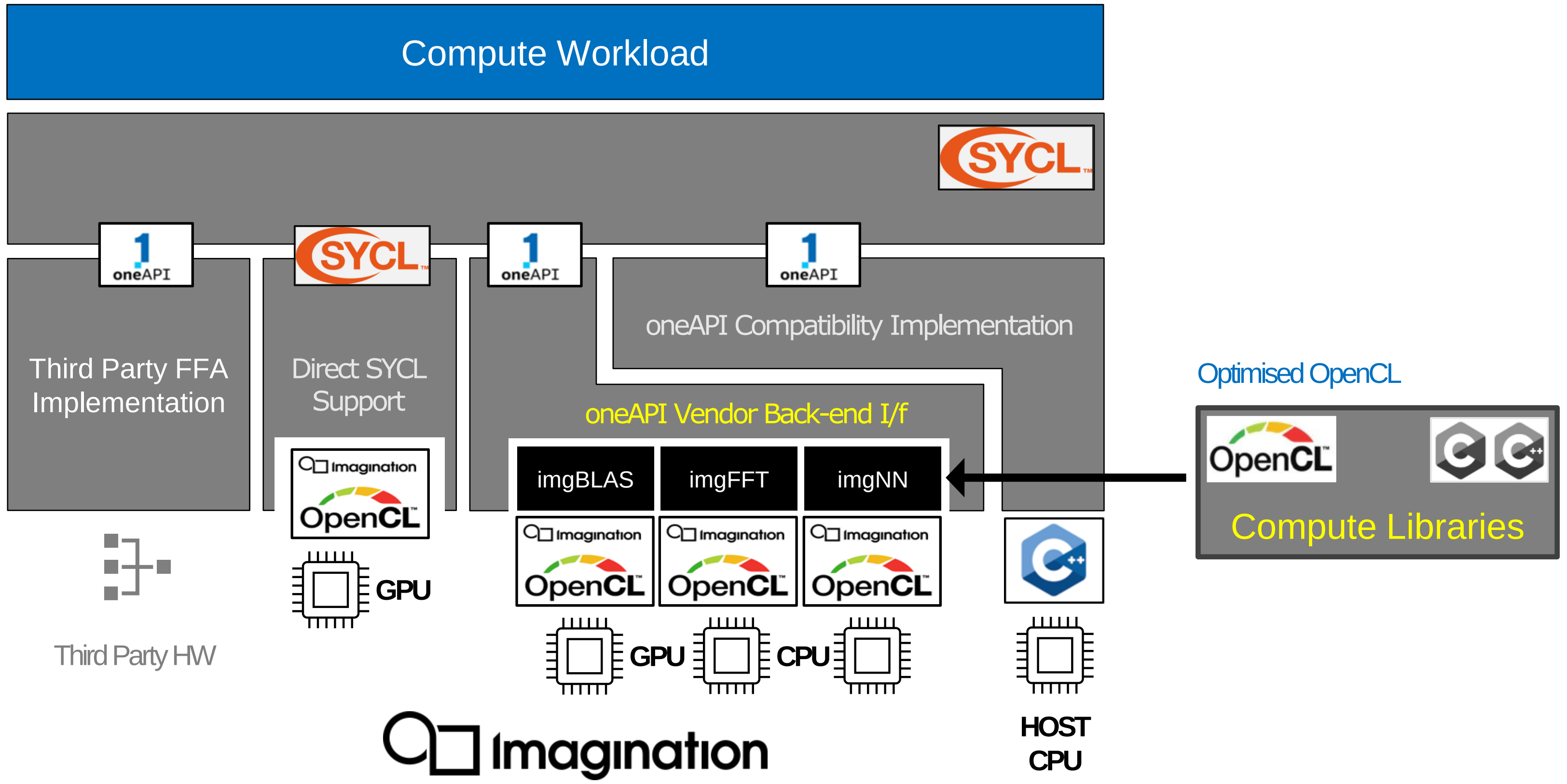
For example:

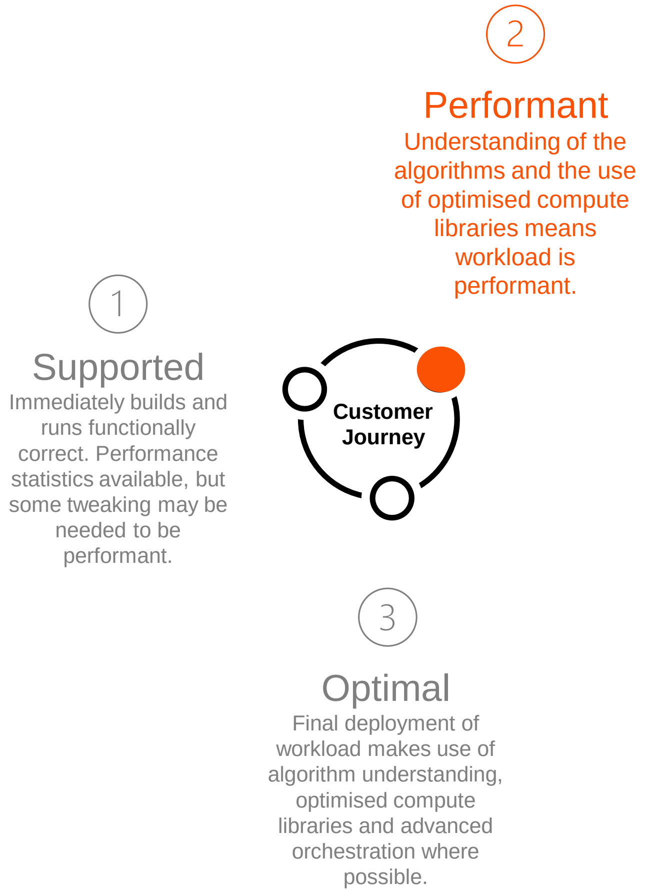
- *imgBLAS ← oneMKL BLAS*
- ***imgFFT ← oneMKL FFT***
- *imgNN ← oneDNN*


cufftPlan1D(…)
Source Code
SYCLomatic
oneAPI & SYCL
plan = dpct::fft::fft_engine::Create(...)
Source Code
oneAPI Vendor Back-end interface
plan = img::fft::Plan1DC2R(…)
Optimised Vendor Library

Hardware Acceleration using oneAPI

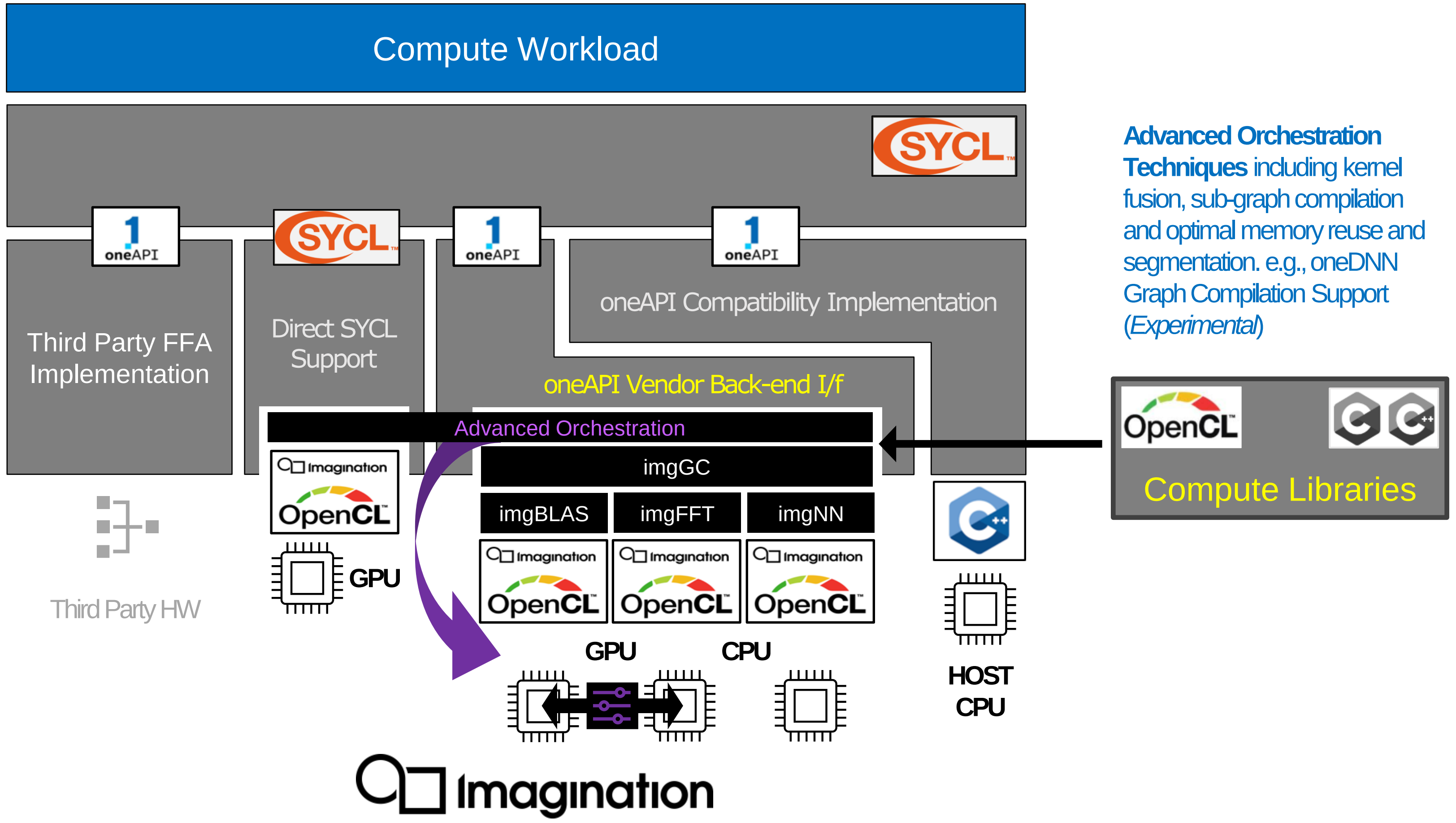
An example of a performant workload using oneAPI vendor backend and optimised compute libraries



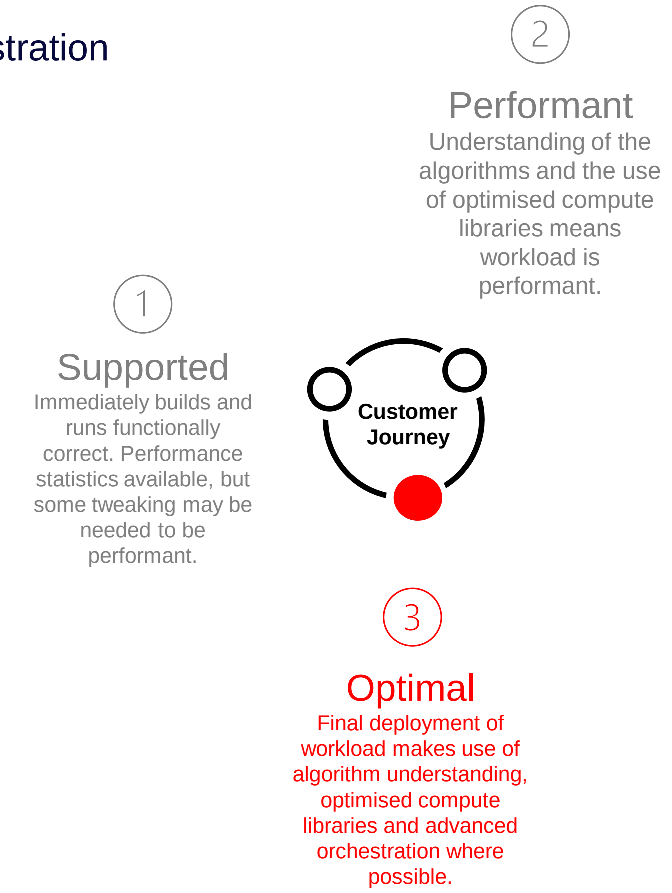

2
Performant
Understanding of the algorithms and the use of optimised compute libraries means workload is performant.
1
Supported
Immediately builds and runs functionally correct. Performance statistics available, but some tweaking may be needed to be performant.
Customer Journey
3
Optimal
Final deployment of workload makes use of algorithm understanding, optimised compute libraries and advanced orchestration where possible.

Hardware Acceleration using oneAPI

An example of an optimal compute workload using optimised compute libraries, GC and advanced orchestration



Advanced Orchestration Techniques including kernel fusion, sub-graph compilation and optimal memory reuse and segmentation. e.g., oneDNN Graph Compilation Support (*Experimental*)



1 Supported
Immediately builds and runs functionally correct. Performance statistics available, but some tweaking may be needed to be performant.

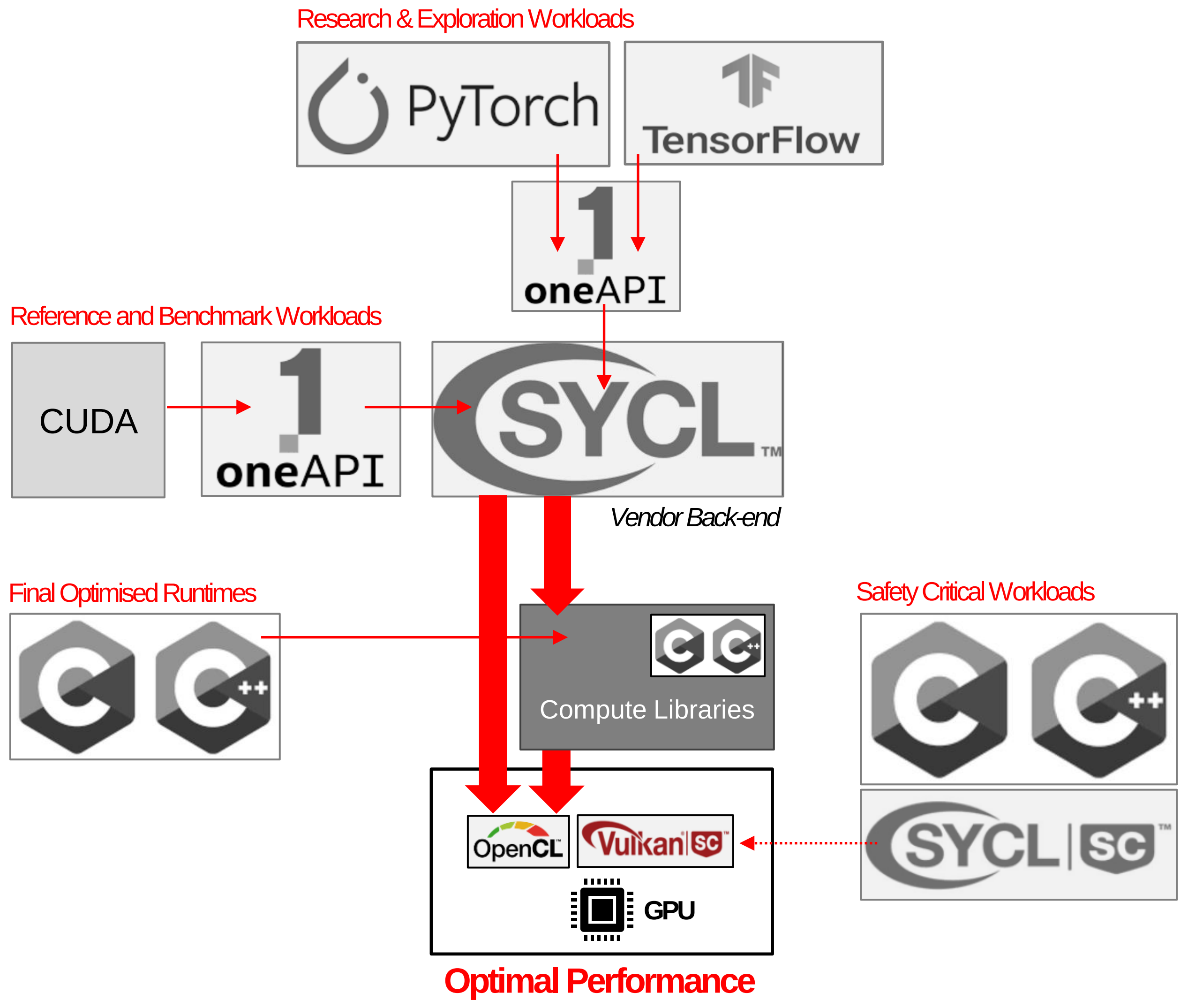
2 Performant
Understanding of the algorithms and the use of optimised compute libraries means workload is performant.

3 Optimal
Final deployment of workload makes use of algorithm understanding, optimised compute libraries and advanced orchestration where possible.

Deployment using SYCL and the oneAPI eco-system

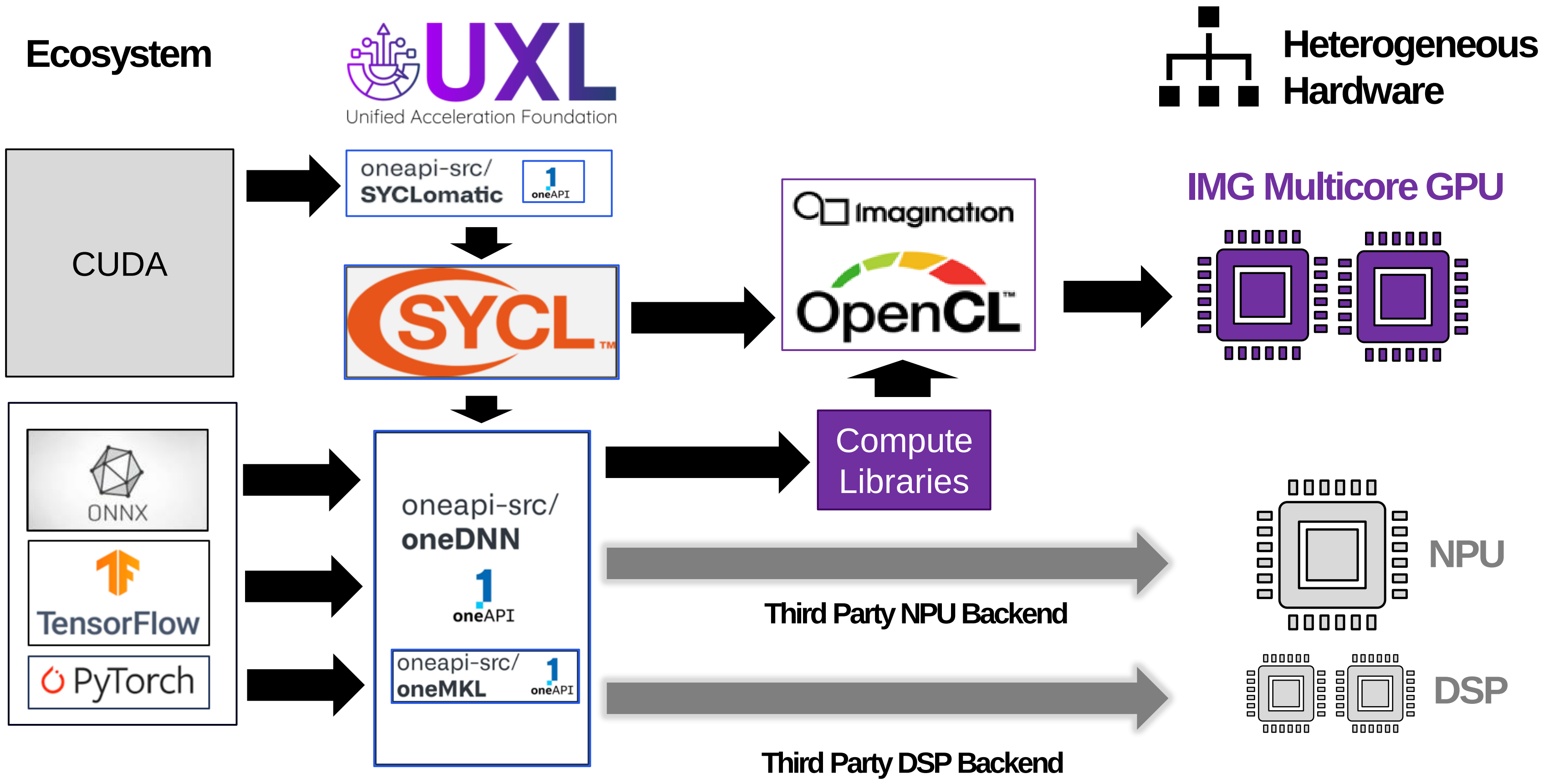
Deployment Pathways for all types of workloads

- **Research & Exploration Workloads**
  - Immediate deployment on GPU and compute IP of SOTA compute algorithms and networks layers.
- **Reference and Benchmark Workloads**
  - Immediate execution of industry standard and recognised workloads; comprehensive KPI's.
- **Safety Critical Workloads**
  - Line of sight for customers when deploying their compute algorithms on our GPU and compute IP in a safety critical domain.
- **Final Optimised Runtimes**
  - Performant, well documented compute libraries and tools to enable customer's success in their software stack creation and deployment.



Multi-vendor heterogeneous computing pathways

Traditional GPGPU and NPU/Graph-Compiled pathways using oneAPI



Summary

- Concept of a customer software development journey is required to be successful in hardware licensing – the oneAPI and SYCL eco-system can facilitate this journey for the customer.
- Conversations about how to migrate existing CUDA workloads are prevalent in customer engagements and this needs to be supported in the development journey.
- SYCLomatic, with its synchronisation to oneAPI libraries can identify cuLibrary calls and convert them to oneMKL libraries which can have optimised "vendor backends" implementations allowing greater control over the structure of kernels that execute on the hardware platform, and hence achieve higher performance.
- Vendor back-end libraries, implemented through proprietary compute libraries can be the most effective way of ensuring performant execution of customer workloads.
- Experimental SYCL extensions can provide the final extraction of optimal performance on your hardware platform
- oneAPI and SYCL allow multi-vendor traditional GPGPU workloads and graph-compiled pathways

2

Performant
Understanding of the algorithms and the use of optimised compute libraries means workload is performant.

1

Supported
Immediately builds and runs functionally correct. Performance statistics available, but some tweaking may be needed to be performant.

Customer Journey

3

Optimal
Final deployment of workload makes use of algorithm understanding, optimised compute libraries and advanced orchestration where possible.

Thank You

UXL

Unified Acceleration Foundation