Date: **December 13, 2012**	**SCOPE STATEMENT FOR EXTREME-SCALE COMPUTING RESEARCH AND DEVELOPMENT (FAST FORWARD) STORAGE AND I/O**

SCOPE STATEMENT FOR EXTREME-SCALE COMPUTING RESEARCH AND DEVELOPMENT (FAST FORWARD) STORAGE AND I/O

LLNS Subcontract No.	B599860
Subcontractor Name	Intel Federal LLC
Subcontractor Address	2200 Mission College Blvd. Santa Clara, CA 95052

Table of Contents

Introduction

The following document satisfies Task 2.1 – Updated Scope Statement of the Extreme-Scale Computing Research and Development (Fast Forward) Storage I/O Subcontract B599860 signed 28JUN2012, assigned to Intel Federal on 25SEP2012, and modified on 27SEP2012.

Problem Statement

Current approaches to HPC software and hardware design will not be sufficient to produce the capabilities required at Exascale. One area of technology that has been identified as a strategic research and development investment that provides benefit to future extreme-scale applications is that of storage and input/output (I/O). Economic realities drive the architecture, performance, and reliability of the hardware that will comprise an Exascale I/O system. Failure will be the norm and the I/O system as a whole will have to handle it as transparently as possible, while providing efficient, sustained, scalable and predicable I/O performance.

Peak performance will come at a premium and specialized subsystems will be required to handle a ratio of burst to sustained I/O rates of at least an order of magnitude. Scalability and fault tolerance will be dominant issues at all levels of the I/O stack and these will in turn impact application design.

Project Goals

1. Research three main areas that cover the whole Exascale I/O Stack from top to bottom

a) Prototype an object-storage API based on HDF5 to support high-level data models, their properties and relationships

b) Prototype the I/O Dispatcher including a burst buffer manager and a data layout optimizer with support for distributed transactions based on PLFS.

c) Prototype Distributed Application Object Storage (DAOS) containers, including support for distributed transactional updates based on the Lustre file system2. Research Big Data – HPC Bridge

2. Research Big Data – HPC Bridge

a) Demonstrate the utility of the I/O stack for potential Exascale-related "Big Data Analytics" applications by using it to store Arbitrary Connected Graphs (ACGs) from commercial off-the-shelf MapReduce systems and run graph computations on them.

b) Develop a "Big Data Graph" generator tool to construct ACGs from large unstructured or semi-structured datasets to exercise the bridge, along with one or more real-world applications that make use of the bridge

In-Scope

HDF

- HDF5 extensions to add support for:
  - End-to-end data integrity
  - Index building, maintenance and query
  - Non-blocking transactional I/O
  - Pointer datatypes to support graph-oriented applications.
- HDF5-based object storage API implemented as a library that can be layered over the I/O Dispatcher API.
- Python wrappers for HDF5 API routines will also be developed to aid ad-hoc query, dataset browsing and debugging.
- Function shipping will be used to export the IOD object storage API from I/O nodes to the compute nodes (CNs).
- Analysis shipping to client environment running on ION and storage server nodes

IOD

- Index construction via client environment running on ION and storage server nodes
- The POSIX API will be supported by the function shipping layer.
- Manage burst buffer caches and leverage PLFS to implement data layout optimization schemas targeting the DAOS API.
- IO Dispatcher (IOD) software running on the IONs based on PLFS plus data migration servers.
- Object versioning, transactions, and key-value stores on the IONs built within IOD
- Aggregate application I/O according to expected usage and implement object placement across DAOS container shards.
- Provide an API to enable higher levels in the stack to explicitly manage the burst buffer and initiate data movements between burst buffer and DAOS.
- IOD layout reorganization designed to work for in-transit data analysis run on the burst buffers as well as pre-staging data for job restart or post-processing analysis which could run either on the compute nodes or on the burst buffers

DAOS

- Leverage many existing Lustre file system components to prototype a fully asynchronous, transactional DAOS API
- Demonstrate that the DAOS API provides a new set of I/O "system calls" with the functionality, performance, scalability, and fault tolerance required to build Exascale I/O systems
- Implement a fault tolerant collective communications subsystem overlaying Lustre servers that allows information to be shared between all servers scalably

- Add support versioning and ultra-lightweight snapshots within storage containers to the Lustre backend storage abstraction, the OSD API, to support transaction rollback and DAOS resource management.

ACG

- Adapt the *GraphBuilder* open source graph construction library (for Hadoop) to the HDF5 API and demonstrate the ingress utility of the I/O stack for the storing of ACGs.
- Adapt the *GraphLab* open source computational framework to the HDF5 API and demonstrate for asynchronous graph-parallel computation on ACGs.
- Provide the HDF5 API with graph partitioning and locality information to load balance the computational effort and minimize communications for ACG applications.

Out of Scope

- Data Security using hierarchical encryption techniques
- Multiple Unlimited dimensions in an HDF5 dataset.
- Resilient burst buffer
- Automated space management on the IONs
- Working with other higher level libraries, such as pnetCDF or ADIOS, to enable them to use IOD/DAOS
- Exploiting the burst buffer NVRAM as a transparent write-back cache for applications that are not burst-buffer aware
- Changes to GraphLab's TCP/IP communications architecture or asynchronous Gather-Apply-Scatter computational model
- Porting of the GraphBuilder library to Exascale beyond its adaptation to HDF5
- Fault tolerance mechanisms for GraphLab or GraphBuilder

Project Constraints

- Unknown future hardware and software limit the Research and Development to current hardware and software capabilities, plus what can be reasonably simulated
- We must be able to run complete stack functionality on single node as well as split ION/CN
- Limitations with current MPI implementations constrain the mechanisms by which a user can run a simulation and a separate analysis job sharing a set of IONs.
- The GraphLab and GraphBuilder open source codebases must remain compatible with commercial off-the-shelf system hardware and software, with the exception of the HDF5 adaptation layer

Project Assumptions

- Application I/O will be objected-oriented, not file-based
- Application I/O will be asynchronous
- The IOD will not be responsible for managing I/O conflicts and for cleaning up after failures of the non-resistant IOD layer
- Applications will use transactional I/O model: all operations in a given transaction will succeed or fail
- HDF5 API interface, with extensions for new functionality, is the entrance to Exascale I/O stack.
- The new DAOS IO system calls will be used in place of the historically used POSIX IO interface.
- The POSIX I/O function shipping in the SOW is oriented toward legacy application support and infrastructure support, like loading shared libraries, etc.
- IONS will be able to communicate horizontally
- The job scheduler will create an allocation that includes CNs and IONs.
- When jobs are run directly on the storage servers, the IOD will be initiated across the storage servers with an MPI communicator just as it is when run on the IONs.

Key Deliverables and Milestones

Quarter 1 – Q3 2012

Deliverables: 1.1 Scope Statement - 30SEP2012

1.2 Detailed Project Plan for Y1 - 31AUG2012

Quarter 2 – Q4 2012

Deliverables: 2.1 Updated Scope Statement – 29DEC2012

2.2 Updated Detailed Project Plan for Y1 – 30DEC2012

2.3 Solution Architecture – 31DEC2012

2.4 Function Shipping Design & Framework Demonstration - 31DEC2012

2.5 DAOS API and DAOS/POSIX Design Document – 31DEC2012

Quarter 3 – Q1 2013

Deliverables: 3.1 Initial Design Documentation – 31MAR2013

3.2 DAOS/POSIX Demonstration – 31MAR2013

3.3 Async I/O and Initial IOD/DAOS VOL Plugin and Function Shipping Demonstration – 31MAR2013

Quarter 4 – Q2 2013

Deliverables:

4.1 Updated Design Documentation – 30JUN2013

4.2 Full IOD/DAOS VOL Plugin and Function Shipping Demonstration – 30JUN2013

4.3 Initial HDF5 Pointer Datatype Demonstration – 30JUN2013

4.4 Data Integrity Demonstration – 30JUN2013

4.5 Reduction Network Discovery Demonstration – 30JUN2013

4.6 Graph Partitioning – 30JUN2013

4.7 Detailed Project Plan for Y2 – 31MAY2013

Quarter 5 – Q3 2013

Deliverables:

5.1 Initial POSIX Function Shipping Demonstration – 30SEP2013

5.2 Object Store Demonstration – 30SEP2013

5.3 N->M Stream Demonstration – 30SEP2013

5.4 DAOS Demonstration & Benchmark Report – 30SEP2013

5.5 Server Collectives/Notification Demonstration – 30SEP2013

5.6 HDF5 Transaction API Demonstration – 30SEP2013

5.7 Full HDF5 Pointer Datatype Demonstration – 30SEP2013

5.8 AAL Demonstration and Benchmark Report – 30SEP2013

5.9 Initial GraphLab/AAL Demonstration – 30SEP2013

5.10 Structure Storage and Retrieval between HDF and IOD – 30SEP2013

5.11 Multi-format Replicas in the IOD – 30SEP2013

Quarter 6 – Q4 2013

Deliverables:

6.1 POSIX Function Shipping Demonstration – 31DEC2013

6.2 HDF5 I/O Dispatcher Object Versioning Demonstration – 31DEC2013

6.3 Basic Analysis Shipping Demonstration – 31DEC2013

6.4 Transactional OSD Demonstration – 31DEC2013

	6.5 Epoch Distribution Demonstration – 31DEC2013
	6.6 Initial Data Analytics Pipeline Integration – 31DEC2013

Quarter 7 – Q1 2014

Deliverables:	7.1 HDF5 Index Plugin API Demonstration – 31MAR2014
	7.2 Burst Buffer Demonstration – 31MAR2014
	7.3 End-to-End Epoch Recovery Demonstration – 31MAR2014
	7.4 Data Analytics Pipeline Integration – 31MAR2014
	7.5 Data Reorganization and Layout Discovery – 31MAR2014

Quarter 8 – Q2 2014

Deliverables:	8.1 HDF5 Index Demonstration – 30JUN2014
	8.2 Advanced Analysis Shipping Demonstration – 30JUN2014
	8.3 Advanced Data Pre-Staging Demonstration – 30JUN2014
	8.4 In-transit Analysis Demonstration – 30JUN2014
	8.5 End-to-End Demonstration with Final Design Documentation and Report – 30JUN2014