**MDTM Project Status (November 2016)**

1. **MDTM Project Background**

DOE’s Office of Advanced Scientific Computing Research (ASCR) funded the Multicore-Aware Data Transfer Middleware (MDTM) project in order to address inefficiencies and limitations with the current generation of data tools running on multicore platforms. MDTM is a collaborative project by FNAL and BNL, with an objective to optimize and accelerate data movement on multicore systems. Multicore and many-core are now the norm in high-performance computing environments. To date, numerous efforts have been made to better exploit these multicore architectures in order to speed up data transfer performance. The current generation of data transfer tools, such as GridFTP and BBCP, have parallelized data transfers in order to provide significant improvement in aggregated throughput, particularly for bulk data movement. Although these parallelization efforts have boosted performance, the existing data transfer tools are still bound by major inefficiencies when running on multicore systems. While there are numerous reasons for these inefficiencies, they fall into two general problem areas: (1) existing data transfer tools are unable to fully and efficiently exploit multicore hardware under the default OS support, especially on NUMA systems; and (2), the disconnect between software and multicore hardware renders network I/O processing on multicore systems inefficient. These inefficiencies are fundamental problems that data movement tools will inevitably encounter when running on multicore systems. Ultimately, these inefficiencies constrain network I/O performance on the end systems. These inefficiencies will become more pronounced as network technology evolves toward terabit networks and 100GE end systems. MDTM is intended to provide a remedy to these inefficiencies.

The MDTM project officially started at Oct 2013. Dr. Wenji Wu of FNAL is the project PI. The project team consisted of FNAL and BNL personnel. Teleconferences were conducted between the two Labs on a weekly basis to discuss progress and technical issues.

* FNAL team: Dr. Wenji Wu (PI), Dr. Liang Zhang, Mr. Phil DeMar, Ms. Lauri Carpenter
* BNL team: Dr. Dantong Yu (Co-PI), Dr. Shudong Jin.

A project website (<http://mdtm.fnal.gov>) was created to help manage the project as well as provide a portal for public outreach.

1. **MDTM Deliverables**

The MDTM project team completed project deliverables on time and within budget. The original project proposal had two deliverables: [1] MDTM middleware, which scheduled platform cores and managed the data transfer threads that ran on them; and [2] a modified version of the BBCP data transfer tool which could effectively utilize the capabilities of the MDTM middleware. As the project design progressed, it became apparent that the single thread per user design of the existing BBCP product and other current generation data transfer tools (GridFTP, etc.) would limit their ability to utilize the MDTM middleware optimally. In order to fully develop and test the MDTM middleware, a very basic FTP-based data transfer software module was developed to drive the MDTM middleware. At later stages of the project, the development tools (mdtmFTP(FNAL) and mdtmBBCP(BNL)) were enhanced to become products. During the course of the MDTM middleware development, it also became clear that instrumentation of the MDTM middleware would be necessary to understand what was happening at the system core/thread level. A GUI-based monitoring capability (mdtmGUI) was developed to provide real-time monitoring data on system core/thread usage. This tool was later enhanced to be a standalone product in its own right. As a consequence, the project team ended up creating and developing four distinct software packages (see Figure 1).

mdtm architecture.pdf

Figure 1 The MDTM project deliverables

Each of these deliverables is described in detail below:

* 1. FNAL deliverables
* *MDTM Middleware*– a user-space resource scheduler that harnesses multicore parallelism to scale data movement toolkits at multicore systems.

MDTM middleware is implemented as a system daemon. Periodically, the daemon collects, monitors, and caches information about the multicore system physical layout (e.g., NUMA topology), configurations, and system loads. Using this information, MDTM middleware will provide query and scheduling services to the data transfer tool, such as mdtmFTP.

Major features include:

* Computer system layout profiling.
* Real-time system status monitoring: (i) CPU usage of each core, and (ii) memory load latency of each NUMA node. This feature allows data transfer tool to use system resources (cores and data buffers) intelligently to avoid overloading particular cores or NUMA nodes.
* NUMA topology-based core scheduling, which supports I/O locality
* Supporting core affinity on I/Os
* System zoning, which partitions system cores into two zones – MDTM zone and non-MDTM zone. Data transfer tool runs in the MDTM zone while other applications are confined to run in the non-MDTM-zone.
* Data buffer allocation and pinning capability.

MDTM middleware was designed to support data transfer tools. However, it can be readily extended to support other types of applications. Additionally, it can be used to study advanced scheduling algorithms and policies on NUMA systems.

* *mdtmFTP*– a high-performance data transfer tool that builds upon the MDTM middleware. mdtmFTP makes uses of some GridFTP modules for rapid prototyping.

Major features include:

* + Adopt a pipelined I/O-centric architecture to execute data transfer tasks. Dedicated I/O threads are spawned to perform network and disk I/O operations.
  + Utilize MDTM middleware to make optimal use of the underlying multicore system.
  + Implements a large virtual file mechanism to address the Lots of Small Files problem.
  + Zero copy, asynchronous I/O, pipelining, batch processing, and buffer pool mechanisms are applied to optimize performance.

Supported file systems:

* + Ext2/3/4
  + XFS
  + Lustre
  + GPFS
* *mdtmGUI* **–** a web-based tool developed to monitor and manage MDTM-enabled DTNs. However, it can also be deployed to monitor and manage any networked computer systems.

Major features include:

* + Online and real-time monitoring of data transfer status and progress
  + Online and real-time monitoring of DTN system status and configurations

These deliverables are all available at: <http://mdtm.fnal.gov>/Releases.html

A Docker release of mdtmFTP is available at: <https://hub.docker.com/r/wenji/mdtm/>

* 1. BNL deliverables
* *mdtmBBCP* – a high-performance data transfer tool that builds upon the MDTM middleware. mdtmBBCP makes use of BBCP modules for rapid prototyping.

Major features include:

* Highly-parallelized and extendable staged event-driven design. This model represents the end-to-end data transfer as a network of processing stages, and each stage is associated with a dedicated thread pool and an incoming task queue. The number of allocated threads in each pool is determined by device’ characteristics and bandwidth capacity.
* Locality-aware resource management. It provides storage-centric task mapping and NUMA-aware thread scheduling to ensure the affinitive movement of data and communications in multicore systems.
* Effective communication protocols for resource allocation, metadata synchronization and payload transfer.
* Various optimization mechanisms, such as zero copy, file sorting, and block-based asynchronous accessing, are applied to improve performance.

Supported file systems:

* Ext2/3/4
* XFS

mdtmBBCP is available at: [https://bitbucket.org/dtyu/mdtmappbin/overview](https://cdcvs.fnal.gov/redmine/projects/mdtm)

1. **Technical Accomplishments**

The MDTM project has been successful on the technical level, both in terms of better performance capabilities for data transfers and in providing new features. On the performance level, mdtmFTP and mdtmBBCP, in conjunction with MDTM middleware, provides superior data transfer performance over existing data transfer tools, particularly within Lots of Small Files (LOSF) scenarios, where both mdtmFTP and mdtmBBCP treat a data set of many files as a single virtual file for data transfer purposes. At the new feature level, mdtmFTP now provides a data streaming capability. In addition, the mdtmGUI provides a real-time systems internals monitoring capability which includes core usage and thread mapping visual displays.

* 1. Performance Enhancements with mdtmFTP & MDTM Middleware (FNAL)

mdtmFTP uses a multi-thread architecture, with separate threads for network I/O, disk I/O, and management functions. Thread scheduling is optimized by pinning threads to cores that provide the best locality for their particular function. Network I/O threads are assigned cores adjacent to their NICs, disk I/O threads are assigned cores adjacent to their disks, etc. In addition, mdtmFTP utilizes multiple optimization mechanisms—zero copy, asynchronous I/O, pipelining, batch processing, and pre-allocated buffer pools—to improve performance. mdtmFTP was evaluated within the ESnet 100GE testbed in comparison to the conventional science data movement transfer tools, GridFTP and BBCP. Testing was conducted between NERSC (Oakland) and StarLight (Chicago), using high performance data transfer nodes (DTNs) with 40GE NICs. The relative transfer times are shown in Tables 1-3. Table 1 shows the results for a single large (100GB) file. Table 2 shows a “folder” transfer with thirty (30) 10GB files. Table 3 shows a folder transfer with a very high number of small files, typical of large science data sets or Internet of Things (IoT) aggregated sensor information. Lower transfer times indicate higher performance.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **mdtmFTP** | **GridFTP** | **BBCP** |
| Time to completion (sec) | 74.18 | 91.18 | <poor> |

Table 1: 100GE file

|  |  |  |  |
| --- | --- | --- | --- |
|  | **mdtmFTP** | **GridFTP** | **BBCP** |
| Time to completion (sec) | 192.19 | 320.17 | <poor> |

Table 2: 30 x 10GE files

|  |  |  |  |
| --- | --- | --- | --- |
|  | **mdtmFTP** | **GridFTP** | **BBCP** |
| Time to completion (sec) | 10.51 | 1006.02 | <poor> |

Table 3: Linux 3.12.21.folder (LOSF)

Most recently, mdtmFTP, with MDTM middleware, was demonstrated at SuperComputing 2016, using 100GE WAN infrastructure between StarLight (Chicago) and the SC16 show floor (Salt Lake City). Using DTNs with 100GE NICs, mdtmFTP achieved ~85Gb/s disk-to-disk.

* 1. New Technical Features/Capabilities of mdtmFTP (FNAL)
* *mdtmFTP Data Streaming:* mdtmFTP has recently been enhanced to include a buffer-to-buffer data streaming capability. At SC16, mdtmFTP was utilized in conjunction with ORNL’s ADIOS data management system to stream fusion simulation workflows from the National SuperComputer Center Singapore (NSCC) to FNAL. The science data streaming was conducted in real-time. Adaptation of other scientific data streaming applications to utilize mdtmFTP is a future objective for the product.
* *mdtmGUI:* mdtmGUI is a web-based tool developed to monitor and manage DTNs that support MDTM middleware. The tool displays real-time system hardware configuration mappings, as well current status of system resources. In addition, the tool displays status and progress of mdtmFTP data transfers, including application thread mapping to system cores. While mdtmGUI was developed to support MDTM middleware monitoring and evaluation, the software package has wider, more generic applicability for monitoring of multicore platforms. The monitoring can be conducted remotely, using a RESTful web interface.
  1. Performance Enhancements with mdtmBBCP & MDTM Middleware (BNL)

mdtmBBCP also uses a multi-thread architecture where dedicated threads are responsible for network I/O, disk I/O, and workload management functions. At the data source, the readers produce loaded buffers, and the senders consume them and then return them to the list of free buffers. After the entire data transfer is completed, the I/O threads are returned to their respective pools for reuse, and a completion event is posted to the upper layer. The I/O threads are designed to be self-suspendable to minimize the CPU consumption while they wait for new tasks. This multi-thread paradigm naturally utilizes multi-core architecture for extremely high performance.

mdtmBBCP is compared with GridFTP, BBCP, and Aspera using the ESNET WAN testbed. For fair comparisons, all the tools are configured with the same parameters—I/O block size and the number of parallel streams. We use Time-to-Completion (TTC) as the performance metric. Table 4 shows the results to transfer a 100GB file from nersc-tbn-2 to nersc-tbn-1. Given the extremely efficient peer-to-peer architecture of mdtmBBCP and simple thread architectures, mdtmBBCP enjoys almost a factor of 2 speed-up over GridFTP, and a factor of 1.7 speed-up over BBCP. We demonstrated this performance in Supercomputing conference at 2015 (<https://youtu.be/1irsHLvxfM8)>.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **mdtmBBCP** | **GridFTP** | **BBCP** | **Aspera** |
| Time to completion (sec) | 55s | 101s | 95s | poor |

Table 4 Comparision among mdtmBBCP, GridFTP, BBCP, and Aspera on large file transfer

Table 5 and 6 show the results of transferring on one Linux kernel folder (3.18.2) and 10 Linux kernel folders in ESNET WAN testbeds, respectively. The “Ratio" values in the tables refer to the speed-up factor of mdtmBBCP over each of the comparative tools. BBCP performs the worst here due to its inefficient protocol design. GridFTP alleviates this drawback by dividing the workloads among multiple FTP sessions and also enables the pipeline mode to transfer multiple files concurrently for each session. In addition, our multi-staged file processing protocol described affords mdtmBBCP a highly efficient mechanism for transferring a massive number of small files within a single FTP session. It not only delivers a 71x to 444.92x speed-up compared over BBCP and GridFTP for the long-haul WAN tests. Here, BBCP requires several days to complete the transfer of ten kernel trees, and as we could not occupy the shared WAN testbed to complete this measurement, we therefore mark “N.A.” for BBCP in Table 6. We also observed that it took mdtmBBCP four seconds longer than Aspera to transfer a single Linux directory in the WAN testbed, while for transferring ten Linux directories, mdtmBBCP achieved a 3.9x speed-up over Aspera.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **BBCP** | **GridFTP** | **Aspera** | **mdtmBBCP** |
| Time To Complete (sec) | 5738 | 1136 | **12** | 16 |
| Improvement Ratio of mdtmBBCP | 358.6 | 71 | **0.8** | 1 |

Table 5: Execution time of transferring one complete set of Linux kernel files over WAN testbed

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **BBCP** | **GridFTP** | **Aspera** | **mdtmBBCP** |
| Time To Complete (sec) | N.A. | 11123 | 98 | **25** |
| Improvement Ratio of mdtmBBCP | N.A. | 444.92 | 3.9 | **1** |

Table 6: Execution time of transferring ten complete set of Linux kernel files over WAN testbed

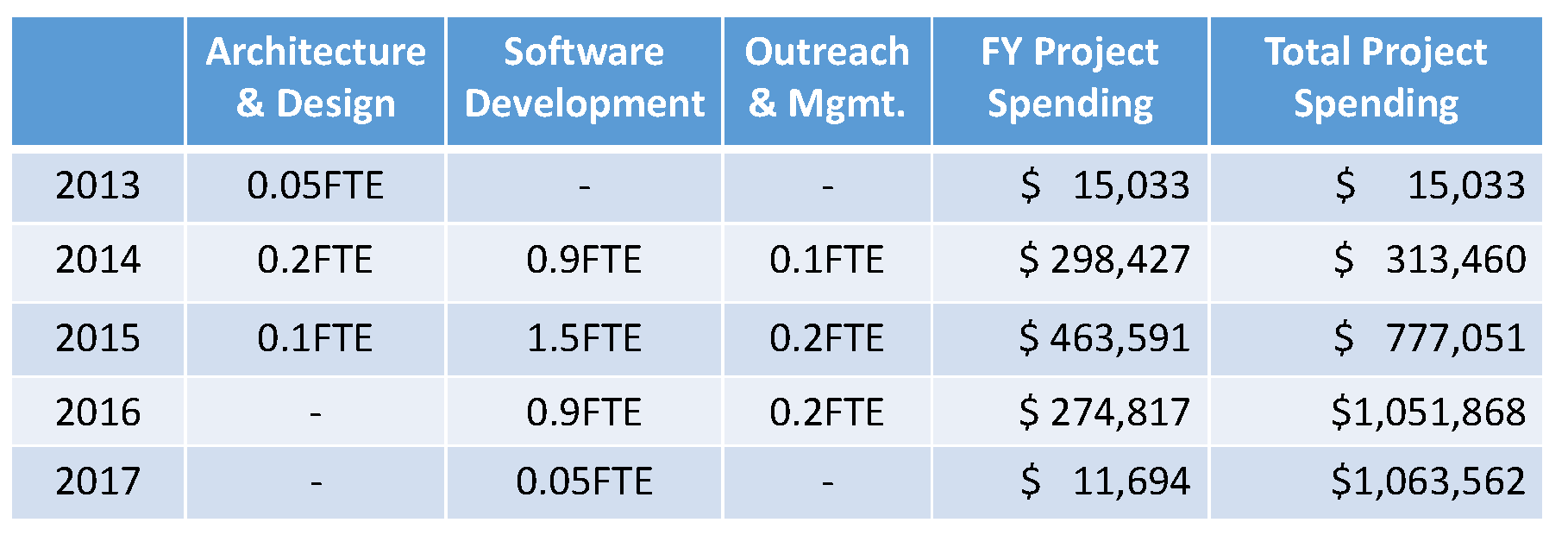


1. **Financial Status:**

The MDTM project was funded as a three year effort, with funding of $350k per year for FNAL, and $200k per year for BNL. The project was started in August 2013, with only a small amount of effort charged within that FY. The bulk of the project’s effort was expended in FY14 through FY16.

* 1. FNAL Component

Project spending for the FNAL component of the MDTM project is shown in Table 4 (below). Initial FY13 effort was limited to a modest level of architecture and design work. The hiring of the principal software developer for the project in late fall of 2013 (FY14) triggered development effort at the level of approximately 1FTE. The project PI contributed roughly 30% of his time for the project in FY14, in a mix of design, project management, and outreach activities. In FY15, a second software developer was brought for roughly 6 months to support the project’s software development efforts. The project PI continued to contribute roughly 30% of his time to the project, with more emphasis on project management & outreach, as design considerations lessened. In FY16, the lead software developer supported the project at a full FTE level until August, when project funding ran down. The project PI provide effort at the 20% level, as he phased his time over to a new, related project. There was a small amount of project funding that carried over into FY17; those funds were spent down in October (2017). The FNAL funding for the MDTM project is essentially spent out at this point.



* 1. BNL Component:



Project spending for the BNL component of the MDTM project is shown in the following table. Initial FY13 effort was limited to a modest level of planning and incurred no spending. The hiring of the research scientist for the project in the beginning of fiscal year 2014 started the design and development effort at the level of approximately 1FTE. The project Co-PI (Dantong Yu) contributed roughly 30% of his time for the project in FY14, in a mix of design, project management, and outreach activities. In FY15, a student software developer was brought for roughly 12 months to support the project’s software development efforts. The project Co-PI continued to contribute roughly 20% of his time to the project, with more emphasis on project management & outreach, as design considerations lessened. In FY16, the research software developer supported the project at a full FTE level until April/01/2016, when the contract with the research scientist expires. The project Co-PI provide effort at the 40% level, as he tries to work with BNL scientist to test and deploy the project. There is a small amount of project funding that carried over into FY17; those funds were spent up in October (2017).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Architecture & Design** | **Software Development** | **Outreach & Mgmt.** | **FY Project Spending** | **Total Project Spending** |
| 2013 |  |  |  | $0 | $0 |
| 2014 | 0.5FTE | 0.5FTE | 0.3FTE | $271,563 | $271,563 |
| 2015 | 0.5FTE | 0.5FTE | 0.2FTE | $211,116 | $482,679 |
| 2016 | 0.2FTE | 0.5FTE | 0.4FTE | $227,384 | $710,063 |
| 2017 | 0.0 FTE | 0.2FTE | 0.2FTE | $90,037 | $800,100 |

1. **Deployment Activities**

5.1 FNAL deployment efforts

mdtmFTP, MDTM middleware, and mdtmGUI have been deployed at the following sites:

* ESNET DTNs
  + CERN site
  + ANL site
  + BNL site
  + LBL site
* HEP community
  + FNAL
  + University of Florida
* Fusion community
  + Singapore ACRC
  + Korean KSTAR
  + PPPL
  + ORNL
  + Stony Brook University

Note 1: our data transfer efforts between KSTAR and PPPL is highlighted and acknowledge by Fusion community.

Note 2: The integration of MDTM with ADIOS is highlighted by OLCF.

<https://www.olcf.ornl.gov/2016/09/28/olcf-fermilab-collaboration-gives-adios-a-boost/>

* StarLight DTNs
* Korean KREONET DTNs

5.2 BNL deployment efforts

mdtmBBCP is deployed at ESNET 100G testbed and BNL site for National Synchrotron Light Source-II (NSLS-II) data transfer.

* ESNET DTNs
  + LBNL ESnet site
  + BNL site

1. **Recent & Ongoing activities**

* FNAL has collaborated with ORNL to integrate MDTM with ADIOS. The goal has been to develop a data streaming capability for mdtmFTP.

This work has been highlighted by OLCF:

<https://www.olcf.ornl.gov/2016/09/28/olcf-fermilab-collaboration-gives-adios-a-boost/>

* FNAL provided two MDTM DEMOs at SC’16
  + Teamed up with StarLight for demo “*mdtmFTP @ 100GE Networks*”
  + Teamed up with PPPL, Brown University, ORNL, ESNET for demo “*Real-time Scientific Data Streaming using ADIOS+mdtFTP*”
  + A data streaming demo movie is available at <http://mdtm.fnal.gov/MDTMUpdates.html>
* A mdtmFTP paper was presented at SC’16 Innovative Networks for Data Intensive Science (INDIS) workshop.
* One Ph.D. graduate student was awarded Standard Performance Evaluation Corporation (SPEC) Distinguished Dissertation Award 2015, <https://research.spec.org/news/single-view/article/winner-of-the-spec-distinguished-dissertation-award-2015.html>.
* Continued effort on hardening mdtmFTP and MDTM middleware to be of production quality, using a combination of testing, debugging, and deployment mechanisms.
* Continued effort on hardening mdtmBBCP and MDTM middleware to be of production quality, using a cobineation of testing, debugging, and deployment mechanisms.
* Work with NERSC data transfer group to deploy mdtmFTP at NERSC.
* Deploy mdtmFTP and mdtmBBCP at more sites and within additional science communities.
* Investigate potential scientific computing use cases for MDTM middleware as multicore platform resource optimization software for other scientific computing applications beyond data transfer.