6-16-2016

Efficient Employment of Large Format Sensor Data Transfer Architectures

Jeffrey R. Oltmanns

EFFICIENT EMPLOYMENT OF LARGE FORMAT SENSOR DATA TRANSFER ARCHITECTURES

THESIS

Jeffrey R. Oltmanns, Civilian, Ctr

AFIT-ENV-MS-16-J-042

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENV-MS-16-J-042

EFFICIENT EMPLOYMENT OF LARGE FORMAT SENSOR DATA TRANSFER ARCHITECTURES

THESIS

Presented to the Faculty

Department of Systems Engineering and Management

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Systems Engineering

Jeffrey R. Oltmanns, BS

June 2016

AFIT-ENV-MS-16-J-042

TITLE

Jeffrey R. Oltmanns, BS

AFIT-ENV-MS-16-J-042

Abstract

Due to the increasing quantity of data collected by Air Force intelligence, surveillance and reconnaissance (ISR) assets and the focus on timely access to the data collected by these systems, operational data transfer network architectures have become a critical component of their employment in the intelligence production process. Efficient utilization of the provided long-haul communications component of the ISR system improves the value of the single asset to the warfighter and enables connectivity of additional assets via the data transfer network architecture. This research effort focused on the creation and implementation of a structured test design methodology based on the principles of Design of Experiments to propose recommendations for optimization of one such operational architecture while avoiding the common pitfalls of inadequate and inefficient test design and implementation. Factors that could influence the performance of the data transfer network architecture were researched and evaluated to recommend the factors of interest that most greatly affect the efficiency of the operational architecture. To support this evaluation, an emulated network testbed was utilized to develop a representative model of system efficiency. The results of this model indicate that increased aggressiveness for data transfer leads to decreased efficiency in the attempt to utilize available network resources, especially in realm of operations under study that represent non-traditional bandwidth delay product (BDP) networks where network delay is the dominating factor in the determination of BDP. The analysis documented a baseline model of system performance that will be used to guide ongoing maintenance, sustainment and enhancement efforts for the current data transfer capability and provides insight into the recommended test design process for use in development and deployment of future capabilities. The ability to model system performance through the use of a structured and straight-forward process allows for the inclusion of the test design and analysis process in software design and development, as well as, system deployment and operations improvements.

Dedicated to Claire, your love of learning, and for continuing to show me the way.

To my family and friends, thank you for your ongoing encouragement and support through all aspects and events of this effort. I could not have finished without you.

Acknowledgments

I would like to thank my research advisor, Dr. Brent Langhals for his mentoring and patience throughout this effort. Your understanding and insight throughout truly helped me to form this effort into a body of work I am proud to put forward. I would like to acknowledge Mr. John Willis for his ongoing efforts on the often referenced "operational system." I look forward to our continued integration efforts and many more test events to further enhance the capability.

Jeffrey R. Oltmanns

Table of Contents

List of Figures

List of Tables

EFFICIENT EMPLOYMENT OF LARGE FORMAT SENSOR DATA TRANSFER ARCHITECTURES

I. Introduction

Background

Current Air Force intelligence, surveillance and reconnaissance (ISR) systems are collecting large quantities of data. Examples of this are wide-area motion imagery (WAMI) and hyperspectral imagery (HSI) sensors. These sensor types collect data on the order of terabytes (TB) per mission. Existing aircraft communications systems do not allow for all of this data to be sent to sensor analysts in real-time due to constraints in the aircraft communication systems. Due to this, data must be stored onboard the collection systems and transferred from onboard storage system to a more sustainable off-board server storage architecture.

At this point in the processing chain, this data is simply that, data that is ready for analysis by trained analysts. To ensure this data is analyzed by the appropriate analysts and thus turned into information the information is stored on this server storage architecture for discovery and access by analysts. If these analysts are stationed where the server storage systems are located, often at a forward operating location (FOL) or forward operating base (FOB), then access and discovery can occur through local networking architectures.

The predominant trend is that of reach back analysis, where the analysts are not stationed at the FOL/FOB but are stationed at a long-term operating location at established bases with established infrastructure for data analysis. By performing this analysis at established locations the required number of forward deployed analysts has dramatically decreased within the current theater of operations.

A specific example of this has been seen in the repositioning of the analysis of WAMI ISR systems. By establishing the ability to perform the processing, exploitation and dissemination (PED) of these systems in a reach back mode of operations forward deployed crews have decreased from 50+ analysts to no forward deployed analysts. HSI systems have capitalized on these lessons learned and deployed no analysts with the ISR system deployment, thus allowing for 75+ analysts to remain at established basing locations and avoid large-scale deployment of the PED infrastructure.

The establishment of a reach back architecture does have its drawbacks, specifically during the forensic or post-mission phases of data exploitation of the sensors. This phase of exploitation requires access to the entire mission data set that is stored on the forward deployed server storage architectures. If the analysts were deployed forward, access to this data would be through local networking infrastructure/architectures. Due to the fact that the analytical portion of the ISR system mission is being performed via reach back, the transfer of the large-scale data sets to the established basing location must be considered.

Due to the geographic separation of the full-mission data set and the analysis teams, the networking architecture that enables processing, exploitation and dissemination (PED) of this data requires utilization of long-haul communications architectures that require unique networking solutions. Utilization of standard data transfer protocols is often times inefficient in the transfer of the large data sets. In direct observations during the transfer of HSI datasets, standard File Transfer Protocol (FTP), which utilizes Transmission Control Protocol (TCP), transmission has been observed to result in a mere 25% network resource utilization thus requiring costly bandwidth expansion requirements to be levied on the communications infrastructure community in order to support timely and efficient transfer of these large datasets. This is as expected as WAN throughput is directly tied to transmission distance and packet loss rate where it can be shown that effective TCP utilization of a single 45 Mbps data flow between a sender and a receiver separated by 1000 miles only nears 30% utilization of total bandwidth resources. Additional performance metrics are documented showing degradation of TCP data transfer from utilization rates of 97% to 32% to 18% of a 45 Mbps network architecture when packet loss rate increases from 0.1% to 3% and 5% respectively (Zhang, Ansari, Wu, & Yu, 2012). This trend is observed throughout academia and industry in numerous publications in Institute of Electrical and Electronics Engineers (IEEE), Association for Computing Machinery (ACM) journals and the establishment of a commercial industry specifically targeted at this very problem set (Dubie, 2004). Additional description of these applications will be described in detail in this document in the Literature Review.

Several approaches exist to improve network utilization statistics to increase utilization of existing bandwidth allocations. One such method requires the establishment of strict quality of service (QoS) metrics that require Layer 2, or link layer, controls for timely delivery of data within the network architecture. Likewise, network transport layer protocols could be developed, tested and fielded to increase optimized utilization of the communications architecture that improve bandwidth estimation of available resources for transmission of data between sending and receiving hosts. As a third option, application layer management techniques may be employed to improve optimization of the same network communications architecture utilization. In current practice within the DoD, the first two options have been administratively removed from the consideration as the entity responsible for operation of these networks has kept that under their strict purview and control. Due to this limitation, current network data transfer architectures for these large-scale post-mission datasets rely upon end-to-end application layer optimization through the use of technologies that do not require modification to the underlying network architecture layers.

Problem Statement

Although network architectures and data transfer capabilities to support transfer of large format sensor data exist, optimization of these systems has required very man-power and resource intensive test and evaluation of both the performance of the network architecture and the data transfer mechanisms that reside within the application layer. Test methodologies utilized to date often follow one-factor-at-a-time test strategies that require strict coordination between the network architecture and sensor data providers in attempts to hold all factors but one constant while attempting to optimize data transfer and reliability rates based on the variation of the "factor du jour" with little consideration for interactions with the constant factors. It is often assumed that a perceived maximum observed response from this methodology will maintain the observed level of performance once the remaining factors are again allowed to vary; however, it has been observed that once departing from a strict test regimen performance is below that which is expected. Unfortunately, due to operational time constraints and a need to get a threshold of capability in operations, a sub-optimum solution is fielded with little to no additional effort extended to reach an objective or optimal solution.

Research Objectives and Investigative Questions

The focus of this research is to perform a thorough analysis of one such operational network architecture and application layer data transfer capability to enable optimized transfer of large format sensor data between the remote and local storage architectures. This analysis will document the variables within the architecture and provide a baseline model for performance of the network architecture for current and future data transfer capabilities. This model will enable timely analysis of the documented factors that affect the data transfer network architecture and demonstrate the ability to develop a structured test design for this problem set to move beyond the time and manpower inefficiencies of one-factor-at-a-time test strategies.

The research documented in this thesis focuses on both academic and commercial optimization techniques utilized for improved data transfer through network architecture and capitalizes on this foundation to determine the true factors that influence the performance of the multi-tiered computer network(s) utilized to transmit data for analysis and production. Through the creation of a representative network architecture, based on modeled performance through network emulation, this research will answer the following questions:

(1) What are the optimal application layer protocol configurations for a specific large-format sensor data transfer architecture?

(2) What are the factors that most greatly affect the performance of this data transfer architecture?

(3) What is the expected performance of the data transfer architecture based upon the developed model for the current employment?

A literature review of related endeavors must first be accomplished to address these questions and establish a realistic level of expectation for the outcomes of any performance model of the network architecture. This review will focus on addressing the following questions:

(1) What are the defining terms in network architecture focused on data transfer?

(2) What defines the specific system architecture that is being reviewed in this research thesis?

(3) What input and control factors are associated with related research or applications that attempt to provide data transfer architecture optimization and how do these factors relate to controllable factors in the specific implementation of data transfer that is the target of this effort?

(4) What are realistic expectations of performance for data transfer optimization capabilities?

Completion of this research effort is also expected to address these additional questions related to test and evaluation efficiencies and operational architecture limitations:

(1) What efficiencies in deployment of future capabilities for optimized data transfer might be realized through the utilization of a structured test design approach?

(2) Is there any justification available to levy additional requirements on the administratively removed sections of the system architecture that would enable further areas of optimization for data transfer network architectures?

Methodology

The previously stated research objectives will be explored through the development of a simulated network that is representative of the specific operational network through the use of an established network emulation capability. Through the use of controlled network emulation, the effects of application-layer factors will be assessed in a development, test and integration environment to adequately model the specific application performance while providing insights into the relationship of the applications operational parameters with respect to data and network parameters. The model developed from this test and evaluation architecture will be analyzed to provide recommendations for operational parameters that affect the performance of the data transfer network architecture.

Assumptions and Limitations

The exact details of the specific operational network architecture are purposefully being omitted from this research. Network performance parameters will be strategically chosen to capture the bounds of performance for the specific operational network architecture to ensure insights gained through this research effort provide relevant insights to the related architecture.

The operational configuration of the network architecture of the existing system is implemented via Layer 3, network layer, controls. While insights may be gained into optimal configuration points that may feed requirements to the service provider, no modifications to this portion of the network architecture will be performed in the emulation of network performance.

This study will not explore communications architecture changes outside of the bounds of the information assurance approval of the operational system. The information assurance approval of the operational system documents specifics of the base operating system and associated applications that are approved for use. This approval, or authority to operate (ATO), establishes the information security posture of the operational system due intensive testing of the system for security vulnerabilities. This resource intensive approval process requires all modifications outside of approved applications to be reviewed and approved for utilization within the system prior to receipt of even an interim authority to test. This directly affects the ability to perform modifications to network protocols or network hardware or associated firmware. An example of this type of modification might involve the development of a modified implementation of a network protocol for this specific implementation. This modification would require changes to the underlying operating system kernel, however, the operating systems on official computers are locked for these type of changes based on the vulnerability analysis performed during the information assurance process for information security purposes. Due to this limitation, variation of factors will be based on the modifications to user-controlled applications that are available within the scope of the approved use of the application within the operational system.

This study will not attempt to make a comparison between different data transmission tools. It will limit its focus to the as-employed network data transfer architecture and end-to-end technical solution. This effort will focus on efficient test design techniques to explore implementation parameters to the existing solution. Future studies might investigate alternative application layer applications and it is assumed that the methodology developed in this effort will lend itself to that comparison testing.

This study will focus on optimization of the data transfer network architecture and assumes that client and server hardware resources are not limited when compared to the data transfer network architecture. Further efforts may investigate optimization of the client and server hardware architecture.

Expected Contributions

The implementation of an optimized data transfer solution for large-format data sensors would provide several advantages to network data transfer mechanisms. Current architectures operate with high operational network overhead to ensure transmission of data in operationally relevant timelines. Through the structured optimization of data transfer mechanisms, overhead can be reduced thus enabling improved utilization of precious network resources.

Development of a structured process through the use of operationally representative network modeling and simulation and/or network emulation could reduce the burden on the operational network for initial configuration and strict control periods for test and evaluation thus providing the large-format sensor data in a more timely manner for intelligence analysis and production. Additionally, with appropriately developed models and assessment methodologies, performance expectations could be more appropriately managed as new network architectures are developed or new/additional large-format sensor programs are employed.

Summary

This introductory chapter outlines the background and motivation for the research to be performed on the optimization of network data transfer of a specific operationally employed network architecture. The defined research objectives and investigative questions help to drive the scope of the research with the associated assumptions and limitations that bound the problem set and planned evaluation. Finally, this chapter highlights expected contributions that will be realized upon completion of the research and analysis.

Chapter II presents the foundational terminology for this effort, related areas of research from both academia and commercial applications that contributes to a thorough understanding of the anticipated performance of the existing data transfer mechanism, as well as, highlighting rationale for the selection of factors of interest as the experimental test design is developed. Chapter III capitalizes on the literature review and understanding of the operational network architecture to derive the research methodology based on statistical design of experiment (DoE) foundations and defines the appropriate system input factors, control variables and response variables for the system under test. This chapter also describes the developed test architecture. Chapter IV provides an analysis of the derived system model and the relationship to the operationally employed network architecture. Finally, Chapter V summarizes the status of the defined research objectives, provides conclusions to the investigative questions and recommends areas of investigation for future research.

II. Literature Review

Chapter Overview

The purpose of this chapter is to document existing research in the areas of network and data transfer optimization and document how insights gained in the various topics relates to an exploration of factors that may have an effect on the response variables of the system under test. The first section defines foundational terminology in the area of network and data transfer. The second section documents areas of related research in network and data transfer optimization of data transfer network architectures with a focus on experimental design applications. The third section defines the definitions of the data transfer network architecture under test. The fourth section describes the philosophies of structured experimental design or design of experiments (DoE).

Definition of Foundational Terminology

Internet Protocol (IP) Stack.

Computer networking is built upon a layered architecture know as the Internet protocol (IP) stack. The IP stack or network protocol stack consists of five layers that provide layer-specific services to enable transmission from a sender to a receiver in a computer network. Data transferred through a computer network transits the network protocol stack during point-to-point or end-to-end transmissions. The five layers of the IP network protocol stack are the application, transport, network, link and physical layers and are shown in Figure 1 (Kurose, 2005).

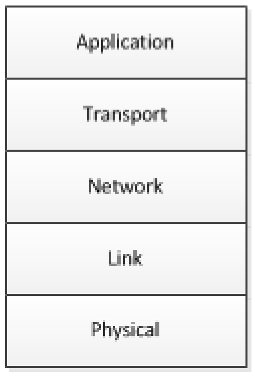


Figure 1 - Internet Protocol (IP) Stack

Each layer of the network protocol stack provides distinct services that enable data to traverse a computer network. Data transmission in the network is enabled by a transfer of data between the layers of the network protocol stack at end and intermediate points within the computer network. The layers of the protocol stack respond to or issue requests to layers that are above or below the specific layer. For example, the physical layer responds to requests from the link layer and the transport layer responds to requests from the application layer and issues requests to the link layer (The LINUX Information Project, 2005). The following sections describe the layers of the network protocol stack and the class of services each layer provides.

Physical Layer.

The physical layer enables the connection of point-to-point nodes within a network by providing the medium in which data is transferred within a network. The responsibility of this layer in the network protocol stack is to transfer individual bits from node to node within the network via the point-to-point specific transmission medium. Examples of a transmission medium are not only physical in nature, such as shielded or unshielded twisted-pair copper wire or single or multi-mode fiber optic cables, but also consist of the wireless medium, such as radio or microwave frequency transmissions. As described by The LINUX Information Project, the physical layer defines the specifications used to interface to the network such as the, “shape and layout of pins in connectors, voltages, cable specifications and broadcast frequencies” (The LINUX Information Project, 2005).

Link Layer.

The link layer provides protocols that utilize the physical layer for point-to-point transmission of data within the network and responds to service requests from the network layer and issues service requests to the physical layer. The specific link layer protocols that are utilized are dependent on the physical medium that is provided by the physical layer. For example, data transferred over a physical medium may utilize the Ethernet protocol as defined in the IEEE 802.3 series of specifications. Data transferred over a wireless medium may utilize the series of wireless local area network (LAN) protocols as defined by the IEEE 802.11 series of specifications.

The level of service provided by the link layer is dependent on the link protocol utilized on the specific node-to-node link (Kurose, 2005). Due to this fact, the link layer may provide varying levels of guaranteed data delivery as specified in the specific link protocol. As the link layer provides only node-to-node delivery services within the network, it is important to consider this fact in the analysis of an end-to-end computer network that may rely on various link layer protocols as data traverses a computer network.

Network Layer.

The network layer provides services to the protocol stack that enable the routing of data transmissions through the network. The network layer responds to service requests from the transport layer and issues requests to the link layer. The network layer is responsible for determination of the series of point-to-point paths that data must traverse to support source to destination end-to-end transmission.

The key component of the network layer is the utilization of the IP protocol as, "all Internet components that have a network layer must run the IP protocol" (Kurose, 2005). The routing of data through the network is determined by specifications of routing protocols, such as, Open Shortest Path First (OSPF) or the Border Gateway Protocol (BGP).

Transport Layer.

The transport layer provides services to the protocol stack that enable end-to-end communication of data across the network. The transport layer responds to service requests from the application layer and issues service requests to the network layer. The transport layer implements various protocols but the two most prevalent for data transfer are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). RFC 793 defines the fundamental design of TCP with subsequent updates, clarifications and modifications found in *Requirements for Internet Hosts -- Communication Layers* (RFC 1122), *TCP Congestion Control* (RFC 5681), *TCP Extensions for High Performance* (RFC 7323), and *TCP Selective Acknowledgment Options* (RFC 2018). RFC 768 defines UDP. The protocols are described in detail in the following sections.

User Datagram Protocol.

The User Datagram Protocol (UDP) provides applications a connectionless service for data transmission and resides within the transport layer. UDP "provides a procedure for application programs to send messages to other programs with a minimum of protocol mechanism. The protocol is transaction oriented, and delivery and duplicate protection are not guaranteed"(Postel J. , 1980). UDP provides no guarantee on delay and no guarantee on in-order message delivery in transmission of data to the application layer.

Due to the connectionless service provided by UDP, it can provide optimizations to application layer requests for service of the transport layer. Four reasons highlighted by Kurose (2005) are as follows:

1. Finer application-level control can be realized due to better understanding of what data is sent from the application and when data is sent from the application. This is due to the fact that as soon as an application process passes data to the transport layer UDP, UDP immediately packages the data inside a UDP segment and then passes the segment to the network layer. Due to this immediate and simple transfer of data, the application knows what data is sent and when.
2. No delay is introduced to establish an end-to-end connection to provide transport-layer reliability. Again, this provides the application layer specific awareness of when to expect transmission of data from the transport layer to the network layer.
3. No connection state is required to be maintained at the transport layer. As no resources are required to maintain this state information, the application layer can typically support more active clients when run over UDP when compared to TCP.
4. UDP requires only a small packet overhead of only 8 bytes per segment transferred. This reduces transport layer overhead in the end-to-end transmission of data over the network.

Due to the amount of packet overhead in the UDP and based on a standard maximum segment size of 1500 bytes, the amount of data that can be transmitted via each UDP segment is 1492 bytes (Kurose, 2005; Postel J. , 1980).

Transmission Control Protocol.

The Transmission Control Protocol (TCP) provides applications a connection-oriented service for data transmission within the transport layer. “[TCP] is intended for use as a highly reliable host-to-host protocol between hosts in packet-switched computer communication networks, and in interconnected systems of such networks”(Postel J. , 1981). The TCP provides three components of service to the application layer: guaranteed delivery, flow control, and congestion control. The implementation of these components is described in the remainder of this section.

Guaranteed Delivery.

Guaranteed delivery relies on the TCP connection-oriented service. The initial step in this guarantee is established prior to transmission of any application-layer data through the exchange of transport-layer control information between the end-to-end sender and receiver. This process is referred to as handshaking (Kurose, 2005).

Guaranteed delivery is also supported by the reliability services as described in RFC 793. Reliability is described as ensuring that, “[the] TCP must recover from data that is damaged, lost, duplicated or delivered out of order” (Postel J. , 1981) by the services provided by the underlying network layer. Reliability is achieved through the transmission of TCP specific transport-layer header information to each message that is received from the application layer. Recovery from data damaged during transmission by the lower layers of the IP stack is detected by the TCP through the use of a checksum for each data segment received. Damaged data is discarded at the transport layer and requested for retransmission prior to delivery to the application layer. Lost data is detected by the TCP through the use of a transmission timeout measurement that is based on the perception of the transport layer interpretation of lower layer performance statistics. If data does not arrive prior to expiration of the timeout interval, the data is assumed to be lost and is retransmitted. Recovery from duplication and out-of-order delivery are achieved through the utilization of a sequence number that is assigned by the sender to each message. A positive acknowledgement (ACK) is transmitted from the receiver to indicate what data has been successfully received and to notify the sender the next data set to transmit. The receiver utilizes the sequence number to correctly order segments and also eliminate duplicate data prior to serving the data to the application layer utilizes the sequence number (Postel J. , 1981; Kurose, 2005).

Flow Control.

Flow control allows the receiver of the data transmission to throttle the amount of data the sender can transmit. This capability is performed through the update of the receive window in the TCP header (Postel J. , 1981). Flow control ensures that resources on the receiver side are not overwhelmed by the transmission rate of the sender (Kurose, 2005).

Congestion Control.

TCP congestion control throttles the amount of outstanding data that a sender can transmit based on perceived congestion within the lower network layers (Kurose, 2005) and is defined in RFC 5681. Congestion control is based on end-to-end feedback at the transport layer because the lower network layers provide no explicit feedback to the end-to-end layer regarding network congestion (Kurose, 2005). Outstanding data is defined as data that has been sent from the sender and has not been acknowledged by the receiving TCP. Four dependent algorithms define the TCP congestion control mechanism: slow start, congestion avoidance, fast retransmit and fast recovery (Allman, Paxson, & Blanton, 2009). Allman explicitly states that there may be occasions when it is beneficial for a sending application utilizing the TCP to be more conservative than the congestion control algorithms allow; however the sending application must not be more aggressive than the algorithms allow (Allman, Paxson, & Blanton, 2009).

Before describing the algorithms that define the TCP congestion control mechanism, it is important to note the internal variables that a TCP connection maintains to manage the use of these algorithms. Allman defines these variables in RFC 5681 as below:

- Slow Start Threshold (*ssthresh*): This variable serves as the delineation between the use of the slow start and congestion avoidance algorithms.
- Receiver Window (*rwnd*): The most recently advertised receiver window that is advertised in the TCP header.
- Congestion Window (*cwnd*): This sender-side state variable limits the amount of data a TCP can send.
- Duplicate Acknowledgement (duplicate ACK): Informs the sender that a segment was received out-of-order and which sequence number is expected. Duplicate ACKs may be caused by several potential network problems. The first may be due to dropped segments by the lower network layers and the second may be caused by re-ordering of data segments within the network.

The slow start algorithm is utilized to ensure the sending TCP gains an understanding of network conditions, as they are unknown prior to transmission. Slow start is utilized to ensure that the sending TCP avoids congesting the network due to transmission of an inappropriately large amount of data (Allman, Paxson, & Blanton, 2009). The slow start algorithm is used at the beginning of a transfer or after repairing a loss due to transmission disruption that exceeds transmission timeout requirements. During this phase, "the TCP sender begins by transmitting at a slow rate but increases its sending rate exponentially until there is a loss event. This algorithm feeds data to the TCP *cwnd* state variable based on feedback received from the TCP receiver and allows it to grow at an exponential rate" (Allman, Paxson, & Blanton, 2009). This allows for timelier probing of the network resources available. Slow start is utilized while *cwnd* is less than *ssthresh*.

"The TCP congestion control algorithm is often referred to as an additive-increase, multiplicative-decrease (AIMD) algorithm. The linear increase phase of TCP's congestion control [algorithm] is known as congestion avoidance" (Kurose, 2005). Congestion avoidance is utilized after the probing efforts of the slow start algorithm and begins a more conservative, linear increase, in the amount of data that be sent into the network. Avoidance of congestion is a primary concern of this algorithm, thus once congestion is perceived, the congestion window is decreased by half when a loss event is detected by the TCP sender.

The fast retransmit algorithm is triggered by the receipt of duplicate ACKs and is used by the TCP sender to detect and repair loss (Allman, Paxson, & Blanton, 2009). This algorithm utilizes the arrival of three duplicate ACKs as an indication of loss of the missing segment due to loss by the underlying network layers. This algorithm receives its name due to the fact that the retransmission of the perceived lost segment is performed prior to the expiration of the TCP retransmission timer (Allman, Paxson, & Blanton, 2009).

The fast recovery algorithm is utilized for the transmissions of data after the initiation of a fast retransmit of a missing segment until the sender receives a non-duplicate ACK from the receiver. The fast recovery algorithm is utilized in this instance rather than the slow start algorithm because it is assumed that the segment has been lost and is no longer consuming network resources (Allman, Paxson, & Blanton, 2009). This is due to the fact that the duplicate ACKs must traverse the network so the network between the sender and receiver is assumed to be intact.

Application Layer.

The application layer provides services to network applications and enables process-to-process communication between end-to-end hosts. Examples of application-layer protocols are the Hypertext Transfer Protocol (HTTP) that enables web document request and transfer and the File Transfer Protocol (FTP) that enables transfer of files between two end systems. Additionally, in the context of the Internet protocol stack, this layer serves as the layer that an end-user application or client-server application resides on for the means of file transfer.

Bandwidth Delay Product.

The inclusion of the high-throughput network architectures drives a relationship to the bandwidth delay product (BDP) of these network architectures. The BDP describes the amount of data an architecture can hold before potential feedback can be received between a sender and receiver. This equates to the number of bits that can exist on the line before actions must be taken by the sending host to avoid congestion of the network architecture. This metric is shown in Equation 1.

$$BDP = b \times d$$

Equation 1 - Bandwidth Delay Product

Where:

b = the bandwidth of the network in bits per second (bps)
d = round trip time (RTT) of the network in seconds (sec)

Due to the relationship of the variables used in calculating the BDP of an architecture, as either variable increases the BDP of the network increases as well. Thus high-BDP network architectures can exist due to increased speed of network transfer or due to increased latency within the network. The operational data transfer network architecture displays characteristics of a high-BDP network due primarily due the moderate network bandwidth combined with the high network latency of the communications pipeline. Due to this increased latency, the ability to utilize techniques such as pipelining, as described in Yildirim (2015) should be considered in approaches aimed at improved optimization of the data transfer network architecture and are explored in this research effort.

Related Research and Experimental Design Observations

This section will document related research in the area of data transfer optimization and experimental design applications. The underlying theme expressed by authors in numerous avenues of research is inefficient utilization of available bandwidth across many varying network architectures. The combination of factors and complications in underlying optimization algorithms throughout the IP stack often create responses that diverge from the optimal. The utilization of TCP for the inherent reliability of data transfer via the protocol has been shown to degrade optimization of network utilization or throughput in many network architectures. This is frequently observed in those network architectures that exhibit performance related to increased bandwidth, increased delay or increased error rates. The simple fact that, “[the] TCP congestion avoidance algorithm interprets packet loss as an indication that the network is congested and that the sender should decrease its transmission rate” yet the actual cause may be due to an simple loss event leads to aggressive reductions in the overall transmission rates observed through standard reliable data transfer means (Hacker, Athey, & Noble, 2001).

These data transfer network architectures tend to focus on transfers of large datasets through network architectures where large datasets are considered on the order of tens of megabytes (MB) to tens of terabytes (TB). Individual file sizes vary in these datasets from what can be defined as small files consisting of tens to hundreds of kilobytes (KB) to large files that can be defined as tens to hundreds of megabytes (MB). The network architectures are often referred to as “high-speed” or “high-bandwidth” where this generally equates to transfer rates of single gigabits per second (Gbps) or higher.

General Methods of Data Transfer.

To ensure reliable delivery of data through a network architecture, some form of feedback must be communicated between the sending and receiving hosts. In this example, a sender must wait for confirmation from the receiver that data was received prior to sending the next packet of information. In high-BDP networks, this method of data transfer leaves the data channel idle while waiting for the receipt of the acknowledgement from the receiver. Downtime in network transfer leads to decreased utilization of the available communications bandwidth and underutilization of available resources provided to allow for data transfer. This non-optimized technique is depicted in Figure 2.

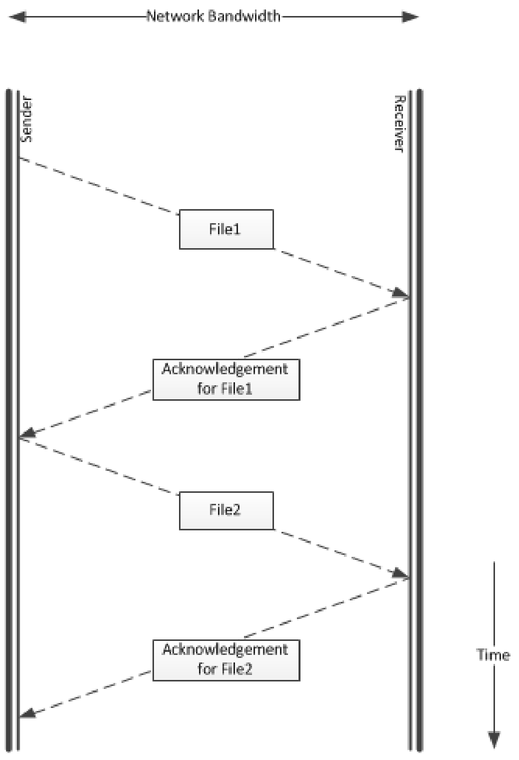


Figure 2 - Non-optimized Data Transfer

The techniques of pipelining, parallelism and concurrency are methods of decreasing this underutilization of existing network resources. These techniques are addressed in an attempt to optimize network utilization in Yildirim (2015).

Pipeline Data Transfer.

Pipeline data transfer in network terms can be related to pipelining in computer architectures where a command is sent through the processing architecture without waiting for an acknowledgement that the previous command has completed. As applied to network architecture, transmission of the next packet of data would be sent from the sender as soon as the previous packet left the sender. The goal of this method is to prevent idle time in the data channel and to minimize delays due to control channel feedback between the sender and receiver (Yildirim, Arslan, Kim, & Kosar, 2015). This technique is depicted in Figure 3. In this method, subsequent files are transferred before the sender is aware that the previous file has been received. Pipelining is utilized in the slow start mechanism within the TCP during its additive increase phase as the sender is able to send unacknowledged packets to the receiver based on the congestion window scaling. Pipeline data transfer targets the problem of sending large quantities of small files (Yildirim, Arslan, Kim, & Kosar, 2015).

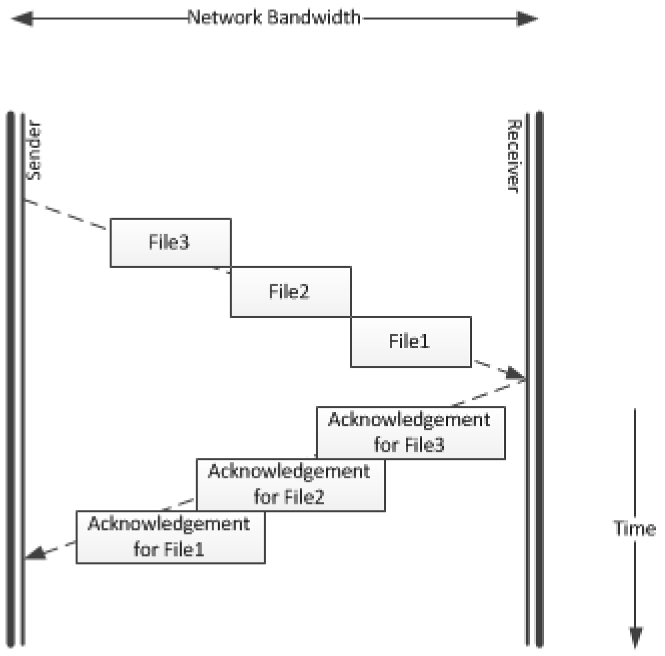

Network Bandwidth
Sender
Receiver
File3
File2
File1
Acknowledgement for File3
Acknowledgement for File2
Acknowledgement for File1
Time

Figure 3 - Pipelined Data Transfer

Parallel Data Transfer.

Parallel data transfer in network terms can be defined as the transmission of multiple segments of the same file via concurrent transport streams. This method divides the available bandwidth between the transmissions of the file segments. Mechanisms utilizing parallel data transfers rely on feedback from control traffic and the performance of the underlying network layers to avoid packet loss due to network congestion. An example of parallel data transfer is shown in Figure 4. Portions of File1 are transmitted in separate transmission streams within the same network architecture. These streams operate independent of each other in a similar fashion to a non-optimized transmission.

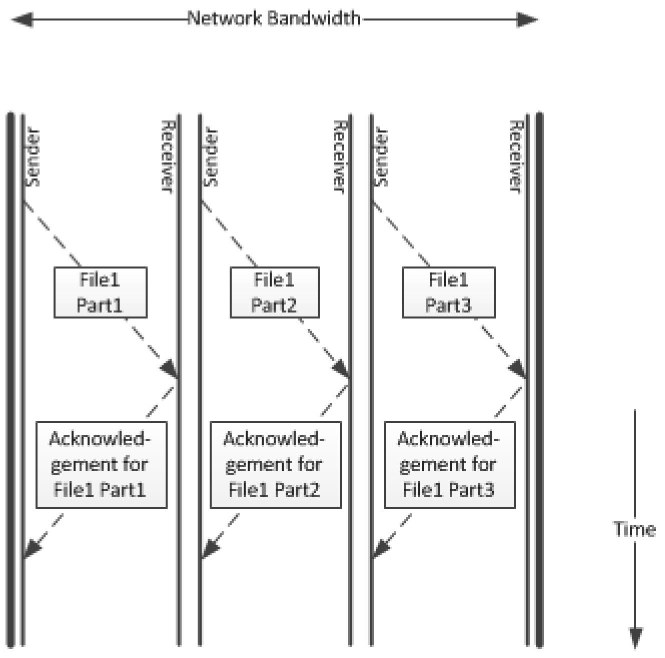


Figure 4 - Parallel Data Transfer

The relationship of packet loss and throughput for a reliable transport TCP stream is shown in Equation 2 and is known as the Mathis equation (Mathis, Semke, Mahdavi, & Ott, 1997).

$$BW = \frac{MSS \times C}{RTT \times \sqrt{p}}$$

Equation 2 - Mathis Equation

Where:

BW = bandwidth of the TCP stream
MSS = maximum segment size which is the largest amount of data that can be sent in a TCP segment (Postel J. , 1981)
C = constant value that is dependent on TCP implementation
RTT = round trip time between sender and receiver
p = number of congestion signals per acknowledged packet (packet loss rate)

Equation 2 can be applied to an estimation of multi-stream TCP bandwidth for *n* parallel streams as shown in Equation 3 (Hacker, Athey, & Noble, 2001).

$$BW_{agg} \leq \frac{C \times MSS}{RTT}\left(\frac{1}{\sqrt{p_1}} + \frac{1}{\sqrt{p_2}} + \cdots + \frac{1}{\sqrt{p_n}}\right)$$

Equation 3 - Parallel Stream Bandwidth

Where:

$p_1 \ldots p_n$ = packet loss rate for each parallel TCP stream
The definition of the remaining factors are defined as in Equation 2

Assuming the parallel streams are receiving equal feedback from the network and are not limited by available network bandwidth the use of *n* number of parallel transport streams can produce increase aggregate throughput by a factor of *n*. This relationship is defined in Equation 4 (Yildirim, Arslan, Kim, & Kosar, 2015).

$$BW_{agg} = n\frac{C \times MSS}{RTT \times \sqrt{p}}$$

Equation 4 - Simplified Parallel Stream Bandwidth

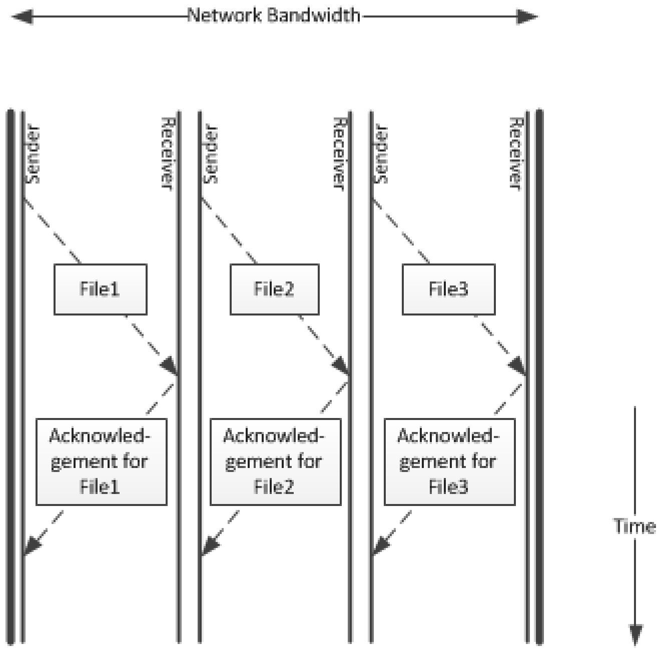
Where:

n = number of parallel TCP streams
The definition of the remaining factors are defined as in Equation 2

In this case it should be noted that the packet loss rate can be random in underutilized networks, however the packet loss rate can increase dramatically when congestion occurs. This initiates the TCP congestion avoidance algorithm which decreases sender throughput based on perceived congestion or observed loss within the network (Yildirim, Arslan, Kim, & Kosar, 2015). Parallel data transfer, as observed by these equations, can only be applied to large data files that allow the optimization mechanism of the transport layer, in this case TCP, to reach a maximum sending rate (Yildirim, Arslan, Kim, & Kosar, 2015). As it relates to this effort, the values of round trip time and packet loss are inversely related to the theoretical throughput available via both parallel and non-parallel data transfers.

Concurrent Data Transfer.

Concurrent data transfer, or concurrency, is defined as sending multiple files through the network simultaneously (Yildirim, Arslan, Kim, & Kosar, 2015). This is similar to parallel data transfer, however, the parallelism refers to portions of a single file being sent in parallel rather than different files being sent in parallel. Concurrency is optimal for transfers of small files and should be considered when attempting to overcome system bottlenecks such as central processing unit (CPU) utilization, network interface controller (NIC) bandwidth and parallel storage system optimizations (Yildirim, Arslan, Kim, & Kosar, 2015). Concurrent data transfer is demonstrated in Figure 5.



Network Bandwidth
Sender
Receiver
Sender
Receiver
Sender
Receiver
File1
File2
File3
Acknowled-gement for File1
Acknowled-gement for File2
Acknowled-gement for File3
Time

Figure 5 - Concurrent Data Transfer

With respect to reliable data transfer it is abundantly clear from research that TCP, while effective in data transfer on networks with the characteristics of high reliability and low latency, does not provide the same optimizations when employed on network architectures operating without these characteristics. The inefficiencies of the methods utilized in TCP for flow and congestion control are often highlighted in research especially when applied to high-speed data networks that display characteristics of a long-fat network (LFN) or high bandwidth-delay product (BDP) (An, Park, Wang, & Cho, 2012; Aspera an IBM(R) Company, 2015; Hacker, Athey, & Noble, 2001; Mathis, Semke, Mahdavi, & Ott, 1997; Miess; Yildirim, Arslan, Kim, & Kosar, 2015)

Conclusions from General Data Transfer.

The relevant observations from (Yildirim, Arslan, Kim, & Kosar, 2015) related to data transfer as they pertain to the techniques of pipelining, parallelization and concurrency for reliable data transfer are captured in Table 1.

Table 1 - Observations from General Data Transfer

Technique	Observations
Pipeline Data Transfer	Targets the problem set of sending large quantities of small files
Parallel Data Transfer	Targets the problem set of sending large data files that allow the optimization mechanism of the transport layer to reach a maximum sending rate
Concurrent Data Transfer	Targets the problem set of sending small files when attempting to overcome system bottlenecks such as central processing unit (CPU) utilization, network interface controller (NIC) bandwidth and parallel storage system optimizations

As pipelined data transfer is utilized in the operational system, it is expected that input factors affecting the general amount of unacknowledged data that is being transferred in the operational system will have an effect on the optimal system response. The observations on parallel data transfer, though not specifically utilized in the operational system, indicate a potential dependency on the ability to optimize the data transfer rate by ensuring the underlying transfer algorithms have sufficient time to optimize their sending transfer rate. The sending and receiving architectures in the operational system are not expected to create a system bottleneck, but the observations on concurrent data transfer could be of interest in a study supporting modification to the system hardware supporting the operational system.

Related Research.

In order to influence the design factors of the experiment, it is important to understand associated areas of research in the field of data transfer optimization. This section will describe several methods of optimization that have proven successful in prior research and application. The intent of this background research is to highlight potential factors of importance within the experimental design phase, determine applicable levels and ranges the parameters in which to perform the experimentation. By understanding multiple research areas that have the same end goal of data transfer optimization, increased confidence in the capture of the experimental space can be gained to ensure critical input parameters are not being overlooked in the experimental design process.

Methods of Network Transfer Optimization.

This section describes research areas that explore different methods of network transfer optimization. In general, these areas can be divided into three focus branches of investigation. The first is the utilization and optimization of TCP-based data transfer to capitalize on the inherent reliability of data transfer with TCP. The second focuses on data transfer via UDP that provides reliable data transfer. The third, less academic method, focuses on optimization of data transfer via commercialized products. The focus of this analysis is to document related test methodologies and test factors as they apply to general optimization of network data transfer and is summarized in Table 2 through Table 12.

TCP Optimization Techniques.

TCP optimization often focuses research in the improvement of the congestion control algorithm or the slow start mechanism. This section describes several capabilities developed for data transfer optimization utilizing TCP and documents observations of experimental design techniques for each effort.

TCP Vegas.

TCP Vegas focuses on three methods to increase throughput and decrease loss in network transmissions. The first method creates a new retransmission mechanism that enables a timelier retransmission of a dropped packet. The next method provides a technique to TCP that enables the ability to adapt the transmission rate based on anticipated congestion within the network thus improving upon congestion avoidance. The final approach focuses on modification of the TCP slow-start algorithm to estimate available bandwidth while avoiding packet loss. TCP Vegas requires a sender-side only modification to the TCP (Barkmo & Peterson, 1995).

Table 2 - TCP Vegas Test Methodology Observations

Protocol: TCP Vegas		
Test Methodologies:	*x*-kernel based simulator	
	Internet-based testing	
Factors:	File Size (KB)	128, 512, 1024
Controls:	Protocol Parameters	Documented decisions for protocol parameters α and β to dampen effect of sporadic changes in perceived network performance
Response Variables:	Throughput (KB/s)	
	Throughput Ratio	
	Retransmissions (KB)	
	Retransmit Ratio	
	Coarse Timeouts	

FAST TCP.

FAST TCP proposes a new congestion control algorithm that is optimized for high-speed, long-latency networks. FAST TCP utilizes four functions to better control data transmission: data control, window control, burstiness control, and estimation. Data control determines which packets to send, window control determines how many packets to send, burstiness control determines when to transmit the packets and the estimation component provides information to the control components on how to make those decisions (Wei, Jin, Low, & Hegde, 2006). FAST TCP requires modifications to the TCP protocol implementation.

Table 3 - FAST TCP Test Methodology Observations

Protocol: FAST TCP		
Test Methodologies:	*dummynet* emulation with *iperf*	(Rizzo) (The Regents of the University of California, through Lawrence Berkeley National Laboratory, 2014)
	ns-2 simulation	(nsnam, 2014)
Factors:	RTT (ms)	50, 100, 150, 200
	TCP Implementation	FAST, Reno, HSTCP, STCP, BIC-TCP
	Number of Concurrent Data Flows	1, 2, 3, 4
	Protocol Parameters	Documented decisions for protocol parameters α and β to dampen effect of sporadic changes in perceived network performance
Controls:	Link layer buffer increase modification	Queue up to 3000 packets to accommodate large packet bursts
	Network Bandwidth (Gbps)	1
	Network Bottleneck Bandwidth (Mbps)	800
Response Variables:	Throughput (Kb/s)	Recorded at receiver via *iperf* sink
	Queue Size (packets)	Recorded on emulated router
	Congestion Window Size (KB)	Recorded at sender
	Observed RTT (ms)	Recorded at sender
	Observed Queueing Delay	Recorded at sender
	Throughput at Bottleneck (Mbps)	Recorded on emulated router
	Number of Lost Packets	Recorded on emulated router

TCP Westwood.

TCP Westwood performs optimizations to the TCP congestion window algorithm focused on wired and wireless network transmissions. The protocol utilizes an end-to-end bandwidth estimation algorithm to discriminate the cause of packet loss within a network between network congestion or physical loss due to network drop. Through the continuous measurement of returning ACKs TCP Westwood performs an estimation of available bandwidth that is utilized in the computation of the congestion window and slow start threshold after a congestion event thus attempting to select these parameters to be consistent with network conditions at the time the network congestion is experienced. This mechanism within TCP Westwood is called faster recovery (Mascolo, Casetti, Gerla, Sanadidi, & Wang, 2001). TCP Westwood requires a sender-side modification to the TCP protocol.

Table 4 - TCP Westwood Test Methodology Observations

Protocol: TCP Westwood		
Test Methodologies:	*ns-2* simulation	(nsnam, 2014)
	Network emulation	
	Network Capacity (Mbps)	10
Factors:	Network Bottleneck Capacity (Mbps)	1, 2, 3, 4, 5, 6, 7, 8
	RTT (ms)	30, 50, 100, 200
	Number of Concurrent Transfers	1, 2, 3 (1 TCP Westwood, 1 or 2 UDP)
	Packet Loss Rate (%)	0.01, 0.1, 0.5, 1 (% packets lost due to error)
	TCP Implementation	TCP Westwood, TCP Reno, TCP SACK
	One Way Propagation Time (ms)	50, 100, 150, 200, 250
Controls:	Packet Size (bytes)	400
	Network Buffers	Intermediate node buffer capacity set to BDP for the scenario
	Protocol Parameter	τ = 500ms *
	UDP Transmission Rate (Mbps)	0 when OFF, 1 when ON
Response Variables:	Throughput (Mbps)	
	Congestion Window (segments)	
	Slow Start Threshold (segments)	
* parameter related to bandwidth estimation sampling which is noted that observed performance is not sensitive to the choice of τ as long as τ > RTT		

Westwood+ TCP.

Westwood+ TCP modifies the bandwidth estimation algorithm to address issues present in acknowledgement compression. Acknowledgement compression is caused by network delay in the delivery of packet acknowledgements from receiver to sender resulting in a clustering of their arrival (Dordal, 2014). Acknowledgement compression allows TCP Westwood to overestimate the available bandwidth causing increased burstiness in network traffic flows utilizing the bandwidth estimation of the protocol. Westwood+ TCP introduces an adaptive decrease algorithm to the congestion control and slow-start threshold variables through the use of filtering and counting the stream of acknowledgment packets to appropriately estimate available bandwidth in the presence of acknowledgement compression (Greico & Mascolo, 2005).

Table 5 - Westwood+ TCP Test Methodology Observations

Protocol: Westwood+ TCP		
Test Methodologies:	*ns-2* simulation	(nsnam, 2014)
Factors:	Network Bottleneck Capacity (Mbps)	10
	RTT (ms)	= 2 + (250/N) where N is the # of concurrent connection up to 252 ms
	Number of Concurrent Connections	10, 20, 40, 60, 80
	Bottleneck Buffer Size (segments)	210
Controls:	Packet Size (bytes)	1500
	Bottleneck Buffer Size (segments)	210
Response Variables:	Throughput (Mbps)	
	Queue Length (bytes)	
	Bandwidth Estimate (b/s)	

TCP-Jersey.

TCP-Jersey performs modifications to the TCP congestion control algorithm by performing available bandwidth estimation and enabling a congestion warning mechanism. Available bandwidth estimation is based on a time-sliding window estimator proposed in Clark and Fang (1998), however, the application of this estimator to the TCP congestion window and slow-start threshold is based upon the congestion warning mechanism (Xu, Tian, & Ansari, 2004). Congestion warning relies on modifications to all in-route routers to mark all packets transiting the hardware when the average queue length exceeds a threshold value of 1/3 of the link buffer capacity (Xu, Tian, & Ansari, 2004). Through explicit understanding of the link conditions that lead to a loss-event, the estimation of available bandwidth of TCP-Jersey offers potential improvement over other estimation techniques. TCP-Jersey requires sender side modifications to the TCP protocol and link layer modifications to network equipment utilized in the transmission path.

Table 6 - TCP-Jersey Test Methodology Observations

Protocol: TCP-Jersey		
Test Methodologies:	*ns-2* simulation	(nsnam, 2014)
Factors:	Network Capacity (Mbps)	1.5, 2, 8, 10, 20, 40, 100
	RTT (ms)	10, 45
	Number of Concurrent Connections	10, 20, 40, 60, 80
	Bottleneck Buffer Size (segments)	210
Controls:	Packet Size (bytes)	1500
	Bottleneck Buffer Size (segments)	210
Response Variables:	Queue Length (packets)	
	Bandwidth Estimate (Mbps)	
	Goodput (Kbps)	

TCP-Cherry.

TCP-Cherry focuses optimization efforts in TCP when utilized with networks containing large propagation delays and therefore large RTT. The protocol utilizes a method of data transfer that probes the network for available resources using data segments that avoid congesting the network. Determination of available resources is performed utilizing low-priority data segments that relay network characteristics but also transmit data. The implementation of TCP-Cherry requires sender side modifications to TCP, inherent support from the link layer for prioritization of data packets, and configuration of all routers in the network path to support generalized head of line priority queuing (Utsumi, Zabir, & Shiratori, 2008).

Table 7 - TCP-Cherry Test Methodology Observations

Protocol: TCP-Cherry		
Test Methodologies:	*ns-2* simulation	(nsnam, 2014)
Factors:	Network Capacity (Mbps)	10, 160
	RTT (ms)	50, 250, 550 (associated with LEO, MEO and GEO satellite delays)
	Probability Error Rate (PER)	10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}*
	Wireless Network Delay (ms)	See above
	Wireless Network Link Error Rate (% packet loss)	See above
	TCP Implementation	Cherry, SimpleRenovated Peach+, Peach+, Hybla, Westwood+, BIC, CUBIC, Compound, NewReno
Controls:	Packet Size (bytes)	1500
	Bottleneck Buffer Size (segments)	210
Response Variables:	Goodput (packets/second)	
	Overhead (%)	
***** PER relates to bit error rate (BER) by PER = 1 - $(1 – \text{BER})^N$ where N is the number of bits per segment. BER of 10^{-10} to 10^{-5} is referred to as very low to high.		

TCP-Illinois.

TCP-Illinois proposes a new congestion control algorithm to optimize TCP transmission over high-bandwidth networks. Packet loss information is utilized to determine whether the transmission window size should be increased or decreased and queuing delay information is utilized to determine how much the transmission window size should be increased or decreased. The protocol replaces the AIMD algorithm of TCP with Concave-AIMD that balances estimated average queuing delay with the amount the transmission window of the sender should be set for increases or decreases. When queuing delay is small, the increase is allowed to be large while decrease is set to the inverse; subsequently the inverse applies when queuing delay is estimated to be large (Liu, Basar, & Srikant, 2008).

Table 8 - TCP-Illinois Test Methodology Observations

Protocol: TCP-Illinois		
Test Methodologies:	*ns-2* simulation	(nsnam, 2014)
Factors:	Network Capacity (Mbps)	10, 40, 100
	RTT (ms)	60, 80, 100, 120
	Concurrent Data Flows	1, 4
	Router Buffer Size (packets)	10, 20, 30, 40, 50, 60, 100, 200
	Probability of Packet Loss	0.0005, 0.001, 0.002, 0.005, 0.01, 0.02, 0.05
	Protocol Parameters	
	Wireless Network Link Error Rate (% packet loss)	
	TCP Implementation	Reno, HS-TCP, BIC-TCP, C-TCP, TCP Vegas
Controls:	Packet Size (bytes)	1000
Response Variables:	Queue Length (bytes)	
	Goodput (Mbps)	

UDP Optimization Techniques.

Due to the connection-less orientation of UDP, optimization techniques that require any level of reliable data transfer using UDP rely on application layer management of data transmission and receipt. Large-scale data transfer of files requires a level of reliability be built in when utilizing UDP for file transfers. Examples of reliable

UDP data transfer technologies are UDP-based Data Transfer (Gu & Grossman, 2007), Tsunami (Miess).

Tsunami

Tsunami is a reliable data transfer protocol that is optimized for improved transfer of large files over high-speed networks. The protocol utilizes UDP for data transfer and TCP for control channel information and replaces the window-based approach of TCP for rate control with an inter-packet delay scheme. Tsunami was written with the intent of advancing experimentation with the protocol, therefore the ability to adjust many of the parameters that relate to protocol optimization are exposed to the user (Miess).

Table 9 - Tsunami Test Methodology Observations

Protocol: Tsunami		
Test Methodologies:	Internet-based testing	
Factors:	Network Capacity (Mbps)	1000
	Network Latency (ms)	0.15, 200
	Size of Data Block	Protocol Parameter
	Target Transfer Rate	Protocol Parameter
	Inter-packet Delay Scaling Factor	Protocol Parameter
	Error Rate Calculation Weighting Factor	Protocol Parameter
	Threshold Error Rate	Protocol Parameter
	Retransmission Queue Size	Protocol Parameter
	Ring Buffer Size	Protocol Parameter
	UDP Send Buffer Size	Protocol Parameter
	UDP Receive Buffer Size	Protocol Parameter
	Update Period Interval	Protocol Parameter
Controls:	Hardware	
	Network	
Response Variables:	Throughput (KB/s)	
	Throughput Ratio	
	Retransmissions (KB)	
	Retransmit Ratio	
	Coarse Timeouts	

The ability of Tsumai to expose algorithm parameters that affect the behavior of the algorithm allows for understanding of parameters in which the author has determined are useful for tuning (Miess). The specific list from Miess is listed below.

1. Which network layer to use (IPv4 or IPv6)
2. The size of each data block
3. The target transfer rate
4. The scaling factors for the inter-packet delay
5. The proportion of historical data used in calculating the error rate
6. The threshold error rate
7. The maximum size of the retransmission queue

8. The maximum number of entries in the ring buffer
9. The size of the UDP send and receive buffers
10. The interval between update periods

Additionally, Meiss notes that the Tsunami protocol does not:

> "[…] attempt to modify global system properties that affect the performance of the protocol, such as filesystem parameters and network interface configuration. [These items] are assumed to be privileged operations outside the scope of the Tsunami application."

This methodology falls directly in-line with the restrictions on architecture modifications in which the operational system must operate per the authority to operate (ATO) of the approved information assurance package.

UDP-based Data Transfer Protocol.

The UDP-based Data Transfer (UDT) protocol is a UDP-based data transfer scheme that attempts to address the problems seen in high bandwidth delay product (BDP) networks. UDT provides an application-level protocol that utilizes the underlying UDP protocol without requiring changes to OS-level configurations. UDT provides reliable data streaming and partial reliable messaging to user developed applications via an application-layer interface (Gu & Grossman, 2007). Due to its utilization in the operational data transfer network architecture, this protocol is described in detail in the section titled UDT Transfer Algorithm on page 54.

Table 10 - UDT Test Methodology Observations

Protocol: UDT		
Test Methodologies:	Internet-based testing	
	Laboratory Testbed	
Factors:	Network Capacity (Mbps)	622, 1000
	Network Latency (ms)	0.04, 15.9, 110
	Concurrent Dataflows	1, 2, 3, 4
Controls:	Hardware	
	UDT Protocol Parameters	
Response Variables:	Throughput (Mbps)	
	Implementation Efficiency (CPU Utilization)	
	Efficiency Index	Metric on how much bandwidth was utilized
	Inter-protocol Fairness	Metric on greediness of protocol
	Stability	Metric on protocol response network change
	Packet Loss	

Commercial Optimization Techniques.

Commercial vendors offer optimized data transfer capabilities that are used to support both DoD and non-DoD applications. While these applications abstract their internal operations, they do allow for user interactions via application layer control. Commercialization of the capabilities supports white paper development and demonstration that provides useful representation of the tools within operational applications and often drives additional areas of research to further increase any perceived competitive advantage. Unfortunately, due to their commercial ties, the underlying application-based protocol technologies lend little insight into the true details of operation for each of the solutions. This section will explore two commercialized tools and attempt to gain insight into representative test parameters from white papers and demonstrations of the capabilities.

FileCatalyst.

The FileCatalyst commercial solution utilizes a UDP-based proprietary protocol that provides accelerated file transfer packaged as commercial solution for enterprise-level file transfers in varying environments (Tkaczewski, 2012). In terms of this research effort, accelerated file transfer as referenced in Tkaczewski (2010) and Tkaczewski (2012) equates to optimized file transfer and is described as such. The FileCatalyst solution operates as an application and attempts to optimize file transfer through a combination of previously described methods. FileCatalyst attempts to capitalize on the connectionless aspects of UDP that do not require in-order delivery or delays in transmission while waiting for acknowledgement of receipt and builds in reliability at the application layer (Tkaczewski, 2010). FileCatalyst also enables concurrent file transfer stream to attempt to eliminate periods of network inactivity. Compression of data is another technique that is utilized to reduce the requirement on network resources by having to send less data across the network (Tkaczewski, 2010). Finally, FileCatalyst enables the capability to perform delta transfers which enables transfer of only the portions of a file that have changed rather than retransmitting the entire file, thus reducing the amount of necessary transfer resources (Tkaczewski, 2010).

Table 11 - FileCatalyst Test Methodology Observations

Protocol: FileCatalyst		
Test Methodologies:	Client-based testing	
Factors:	Network Capacity (Mbps)	10, 50, 100, 10000
	RTT (ms)	100, 160, 290
	Packet Loss (%)	0, 1
	Transfer Protocol	FTP, FileCatalyst
Controls:	FileCatalyst Application Parameters	
Response Variables:	Throughput (Mbps)	
	Bandwidth Utilization (% of available)	

Aspera.

The Aspera commercial solution is based upon the patented Fast and Secure Protocol (FASP™) transport technology that is targeted for the high-speed movement of large files or collections of files over wide area networks (WANs) (Aspera an IBM(R) Company, 2015). FASP™ is implemented at the application layer and as such does not require any changes to underlying network layers for utilization. FASP™ uses standard UDP transport layer transmission and decouples congestion and reliability control in the application layer (Aspera an IBM(R) Company, 2015). Rate control is performed via a delay-based mechanism that adapts based on measured queuing delay with a proportional congestion control mechanism (Aspera an IBM(R) Company, 2015). FASP™ attempts to maintain a small and stable amount of queuing in the network and proportionally adjusts transfer rate based on the difference between the measured queuing delay and the target queuing delay (Aspera an IBM(R) Company, 2015). In discussions on the performance of the protocol, the vendor introduces the metrics of sending cost and receiving cost that are described in Equation 5 and Equation 6 respectively. The discussion of these metrics attempts to describe the efficiency of a protocol to transfer file data in comparison to the amount of resources utilized to complete a successful file transfer and is similar in nature to the concept of goodput.

$$sending\ cost = \frac{total\ bytes\ sent - actual\ bytes\ received}{actual\ bytes\ received}$$

Equation 5 - Protocol Sending Cost

Where:

total bytes sent = the number of bytes transmitted from a sending client
actual bytes received = the number of bytes received by the receiving client

Sending cost equates to network loss due to packet loss from network congestion (Aspera an IBM(R) Company, 2015).

$$receiving\ cost = \frac{total\ bytes\ received - actual\ useful\ bytes}{actual\ useful\ bytes}$$

Equation 6 - Protocol Receiving Cost

Where:

total bytes received = the number of bytes received by the receiving client
actual useful bytes = the number of bytes received that were the payload of the data transfer

Receiving cost equates to the amount of duplicate payload information received that is provided due to duplicate retransmissions from the sending client (Aspera an IBM(R) Company, 2015).

Table 12 - Aspera FASP(TM) Test Methodology Observations

Protocol: Aspera FASP™		
Test Methodologies:	Unspecific Internet-based testing	
Factors:	Network Capacity (Mbps)	100, 155 (OC-3), 300
	RTT (ms)	20, 40, 50, 60, 80, 100, 120, 140, 160, 180, 200, 400, 800
	Packet Loss (%)	0, 0.05, 0.1, 1, 2, 5, 10
	Transfer Protocol	FASP™, FTP, TCP Reno, FAST TCP, UDT
Controls:	Protocol Parameters	
Response Variables:	Throughput (Mbps)	
	Goodput (Mbps)	
	Sending Cost	As defined in Equation 5
	Receiving Cost	As defined in Equation 6

Conclusions from Related Research.

The study of related research on network data transfer optimization allows for an understanding of test methods utilized for demonstration of capability in this area. Understanding the scope of parameters, or factors, utilized in these testing scenarios plays a significant role in determining optimal configurations for testing to capture the realm of performance for the operational data transfer network system architecture under test. As was discussed in this section, testing methodologies varied from pure modeling and simulation of the network architectures utilizing network simulation and emulation tools such as ns-2 (nsnam, 2014) and WANem (Nambiar, et al., 2014) to testing on networks that included academic, research and development, and operational networks supporting large data transfer for scientific and commercial uses. These methodologies are summarized in Table 13.

Table 13 - Test Method Summary

Test Method	Research Effort
General Simulation	TCP Vegas
ns-2	FAST TCP, TCP Westwood, Westwood+ TCP, TCP-Jersey, TCP-Cherry, TCP-Illinois
Network Emulation	FAST TCP, TCP Westwood
Laboratory Testing/Client-based Testbed	UDP-based Data Transfer Protocol, FileCatalyst
Internet-based Testing	Tsunami, UDP-based Data Transfer Protocol, Aspera

The study of related research of network data transfer optimization also allows for insights into test factors for consideration when testing optimization techniques or applications. Common factors tested through the studies were network parameters, such as, bandwidth, latency and reliability. The range of factors tested is summarized in Table 14 and can be used to validate selection of test factors for the operational network architecture under test.

Table 14 - Factor Settings Summary

Factor	Factor Settings		Research Effort
	Low	High	
Network Bandwidth	1.5 Mbps	1 Gbps	**Low:** TCP-Jersey **High:** Tsunami, UDP-based Data Transfer Protocol
Network Latency	0.04 ms	800 ms	**Low:** UDP-based Data Transfer Protocol **High:** Aspera
Network Reliability	0%	10%	**Low:** FileCatalyst ,Aspera **High:** Aspera
Algorithm Packet Size	400 bytes	1500 bytes	**Low:** TCP Westwood **High:** Westwood+ TCP**,** TCP-Jersey**,** TCP-Cherry

Finally, the study of related research highlighted response variables of interest for test and evaluation of optimization of network performance. Response variables of interest were throughput, observed latency, specific protocol estimation parameters (i.e. estimated bandwidth, estimated congestion window), network queue lengths, fairness, goodput, overhead and overall utilization rates. Additionally, response metrics that can be related to specific response variables were discussed, such as the sending and receiving cost. The general response variables utilized in the different tests are summarized in Table 15.

Table 15 - Response Variable Summary

Response Variable	Definition from Related Research	Research Effort
Throughput	The raw transfer data rate achieved by the research effort between sender and receiver. This includes all protocol overhead and retransmissions of data.	TCP Vegas, FAST TCP, TCP Westwood, Westwood+ TCP, Tsunami, UDP-based Data Transfer Protocol, FileCatalyst, TCP-Jersey, Aspera
Goodput	The receiving rate of actual data inside the transferred packets that is the target of the data transfer; "…the effective amount of data delivered through the network" (Xu, Tian, & Ansari, 2004).	TCP-Jersey, TCP-Cherry, TCP-Illinois, Aspera
Utilization	A metric to describe the rate of consumption of available network resources from the transfer of the data. Often this is based on the ratio of throughput to network bandwidth but could also be the ratio of goodput to network bandwidth.	UDP-based Data Transfer Protocol, FileCatalyst
Sending Cost	A metric developed to depict network loss due to congestion.	Aspera
Receiving Cost	A metric developed to depict the amount of retransmissions required to successfully transfer the data file.	Aspera

The data gathered in Table 13, Table 14, and Table 15 provides the basis for creation of the test design of the network architecture under test which is described in the following section. By understanding the available test methodologies, ranges of factor inputs and relative response variables associated with test and evaluation of network optimization techniques, the selection of the parameters in the test design ensures a thorough understanding of the factors and ranges of interest in both academic and practical operation of data transfer network architecture test and operations. Due to the utilization of response variables of interest that are based on related testing, conclusions and comparisons from this effort can be drawn based on similar understanding of terminology.

Definition of the Network Architecture under Test

This section describes the details of the system under test as they relate to the specific operational data transfer network. As described in the Assumptions and Limitations section of Chapter II, the details of the specific operational network architecture are purposefully being omitted from this research. Network performance parameters will be strategically chosen to capture the bounds of performance for the operational network architecture to ensure insights gained through this effort provide relevant insights to the related architecture. Additionally, the bounds of the information assurance approval, or authority to operate (ATO), limit the ability to perform modifications to network protocols or network hardware and/or associated firmware. This section describes the linkages between the operational system architecture and the data transfer network architecture under test.

Operational View.

The operational data transfer network architecture supports the transmission of large data sets between data storage located at a fixed base location and data storage located at a forward operating location. The general performance of the data transfer network is based upon requirements levied upon the service provider to support a specific quantity of reach back data transfer between the forward operating location and the fixed base location. At each site, a data interface server is established to perform the actions of a sender/receiver based relationship that supports the transfer of the sensor data from the forward operating location to the fixed base location. This relationship is shown in the high-level operational concept graphic in Figure 6.

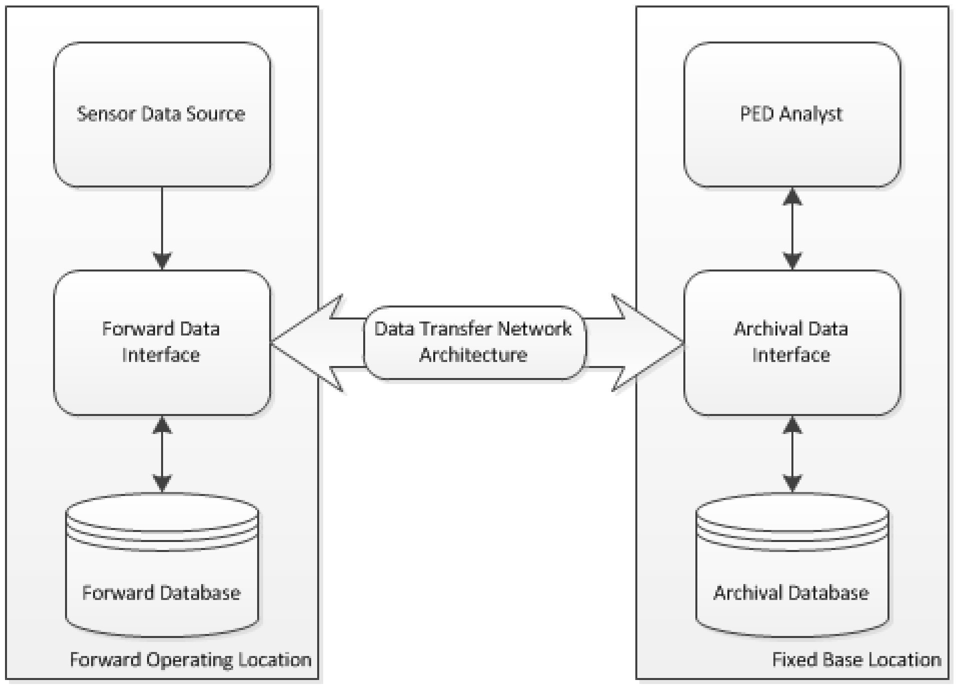


Figure 6 - Operational View of System Architecture

In this system, data is transferred to a forward database for local storage via a forward data interface. This interface controls ingestion of the data into the forward database for future transmission from the forward operating location to the fixed base location. Resources available to the both the forward data interface and the archival data interface are sufficient to ensure that processing, storage and retrieval actions do not provide an artificial limitation on the ability to transmit and receive data from the data transfer network architecture. The network optimization application executes on both of these interfaces. The operational data transfer network architecture provides a pseudo-guaranteed transmission rate based upon the requirements levied to the procuring organization, however, these requirements only establish a point-to-point connection between the archival data interface and the forward data interface at the transport level.

Operations at the link and physical layers are governed by the providing organization with limited quality of service (QoS) attributes. These QoS attributes are likely driven by network capacity and network path or routing but these items are only apparent to the forward and archival data interfaces by perceived latency in the network. By receiving only this feedback response from the network, the data transfer optimization application can only estimate the true cause of increased or decrease network latency and must make appropriate modifications to transmission behavior with insufficient knowledge. Where this comes into play is the fact that increased latency could be due to over-utilization of the available network resources, congestion, or it could be due to degraded performance of the overall network via a service outage or managed routing that utilizes internal links with varying levels of inherent latency.

An example of this routing example might involve the use of satellite communications that have inherently longer transmission timelines than equivalent terrestrial communications. In fact, as seen in testing for TCP-Cherry in Table 7, the use of different communications satellites between the LEO, MEO and GEO belts can provide drastic differences in latency for a communications system (Utsumi, Zabir, & Shiratori, 2008).

Network Optimization Algorithm.

The data transfer mechanism between the forward data interface and the archival data interface is based upon utilization of the UDP-based Data Transfer algorithm that was introduced earlier in the UDP-based Data Transfer Protocol section. The following section builds upon the overview of UDT and provides detailed documentation on the parameters of the UDT algorithm and those available in the operational instance. These parameters are exposed to the data interfaces at each end of the operational data transfer network architecture and allow for modification during transmission of the large format sensor data as from the forward operating location to the fixed base location.

UDT Transfer Algorithm.

The UDT protocol "addresses the problem of transferring large volumetric datasets over high bandwidth-delay product (BDP) networks" via a UDP-based approach that employs congestion control techniques focused on networks that support other applications and protocols (Gu & Grossman, 2007). UDT creates a sender-receiver relationship between two hosts where data is sent from a sender to a receiver, whereas, control flow messages are sent between the receivers of each host (Gu & Grossman, 2007). The following sections highlight the documented algorithms utilized within the UDT protocol for rate and flow control.

Rate control in the UDT protocol is performed via a rate-based congestion control while flow control is performed via a window-based process to regulate data traffic from the sender (Gu & Grossman, 2007).

UDT Congestion Control.

UDT relies upon a rate-based congestion control and a window-based flow control to govern data transferred from the sender to the receiver (Gu & Grossman, 2007). “Rate control updates the packet-sending period every constant interval, whereas flow control updates the flow window size each time an acknowledgement packet is received” (Gu & Grossman, 2007).

UDT Transmission Rate Control.

UDT rate control is performed on a constant interval, referred to as the synchronization time interval (SYN). By default this factor is set to 0.01 seconds. UDT rate control performs in an additive increase multiplicative decrease (AIMD) methodology that is referred to as decreasing additive increase multiplicative decrease (DAIMD) and relates to an AIMD algorithm as AIMD with decreasing increases. This is performed to quickly increase sending rate but as the sending rate approaches the perceived bandwidth of the network, the increases are not as dramatic. Performing in this manner attempts to decrease the effect of estimation errors in the bandwidth estimation process (Gu & Grossman, 2007).

Decreasing Additive Increase.

UDT increases the packet-sending rate every rate control interval in which there are acknowledgements received and no negative feedback from the receiving host that indicate loss or increasing delay via negative acknowledgements or timeouts. The rate control interval, of UDT is 0.01 seconds(Gu & Grossman, 2007). The sending rate is governed by the formulas shown in Equation 7 and Equation 8 (Gu & Grossman, 2007).

$$x \leftarrow x + \alpha(x)$$

Equation 7 - UDT Additive Increase

Where:

x = the sending rate
and

$$\alpha(x) = 10^{\lceil \log(L - C(x)) \rceil - \tau} \times \frac{1500}{S} \times \frac{1}{SYN}$$

Equation 8 - UDT DAIMD Algorithm

Where:
L = estimated link capacity measured in bits/second as shown in Equation 10
SYN = the fixed rate control interval of UDT, or synchronization time interval, which is 0.01 seconds
S = UDT packet size (in terms of the IP payload) in bytes
C(x) = function that converts the unit of the current sending rate, x, from packets/second to bits/second (*C(x) = x * S * 8*)
τ = UDT protocol parameter, which is 9 in the protocol specification
1500 relates to 1500 bytes which is treated as the standard packet size

Multiplicative Decrease.

UDT attempts to differentiate between packet loss due to congestion and loss due to error the use of a negative acknowledgement (NAK) which is sent from the receiver to the sender to indicate a loss event (Gu & Grossman, 2007) A congestion event is a specific instance of a loss event where, “the largest sequence number of the lost packets in [the] loss event is greater than then largest sequence number that has been sent when the last rate decrease occurred” (Gu & Grossman, 2007). The decrease in sending rate is governed by Equation 9 (Gu, Hong, & Grossman, 2004).

$$x \leftarrow (1 - \beta) \times x$$

Equation 9 - UDT Multiplicative Decrease

Where:
x = the sending rate
β = a constant decrease factor such that $0 < \beta < 1$; defined in UDT as 1/9

To avoid reducing packet sending rate due to non-congestion event losses such as link or physical layer errors, UDT does not react to the first packet loss in a loss event (Gu & Grossman, 2007). This methodology attempts to avoid unnecessary reduction in sender transmission rate when bandwidth is actually available.

Bandwidth Estimation.

Bandwidth estimation is performed by UDT using receiver-based packet pairs to perform an estimate of the current bandwidth. This is performed through the use of a packet pair that is sent with the omission of the inter-packet waiting time every 16 data packets (Gu & Grossman, 2007). The link capacity is estimated by Equation 10 (Gu & Grossman, 2007).

$$L = \frac{S}{T}$$

Equation 10 - UDT Bandwidth Estimation

Where:

S = average packet size of the packet pairs in bits
T = median inter-arrival time of the packet pairs in seconds

UDT Flow Control.

UDT flow control governs the amount of data that a sender can transmit without overwhelming the receiver and is implemented on a window based algorithm (Gu & Grossman, 2007). Flow control limits the number of unacknowledged packets that the sender can transmit and is controlled by the receiver during data acknowledgments. This control mechanism occurs every SYN time and updates the sender transmit window to be the minimum of the transmit window size, as defined in Equation 11 (Gu, Hong, & Grossman, 2004), and the available receiver buffer size.

$$w = w \times \lambda + AS \times (SYN + RTT) \times (1 - \lambda)$$

Equation 11 - Sender Receive Window Size

Where:

w = sender transmit window size in packets
λ = factor for the moving average such that $0 < \lambda < 1$
AS = packet arrive speed since last time w is updated*
SYN = UDT SYN period
RTT = network round trip time

**w* will not be updated if no packets arrive during the SYN period or there are too few packets to estimate the arrival speed (Gu, Hong, & Grossman, 2004)

Factors of UDT.

Synchronization time interval (SYN): By default this factor is set to 0.01 seconds and serves at the fixed rate control interval for the protocol (Gu & Grossman, 2007). Rate control within UDT is performed on this constant interval. This interval is user selectable. UDT generates acknowledgements at a fixed interval. Due to this, updates to factors that are dependent on this acknowledgement rate can be related to the bandwidth and RTT of the network in use. If the bandwidth is faster then there is less control information being sent in the network as compared to the amount of data that is being transferred. However, if there is a high RTT or lower transfer rate, then the amount of control information to the data rises (Gu & Grossman, 2007).

In-order delivery: UDT supports buffering of arriving packets to provide in-order delivery should it be required of the application. When in-order delivery of packets is required a packet cannot be delivered to the application until all packets prior to it are either delivered or dropped (Gu & Grossman, 2007). This factor is user selectable.

UDT Maximum Segment Size: User definable factor that controls the packet size for UDT. Default is 1500 bytes.

Understanding of the expected number of packets to be sent during a UDT ACK interval provides insight into the amount of data being transmitted via a network in relationship to the amount of control and feedback information being generated. This provides insight into the amount of pipelined data transfer that is occurring within the UDT application protocol.

$$NumberPackets_{expected} = \frac{DataRate_{expected} \times ACKinterval}{PacketSize}$$

Equation 12 - Packet Transmissions per ACK Interval

Where:
NumberPackets$_{expected}$ = number of packets received per ACK interval
DataRate$_{expected}$ = expected transmission rate of the network in bits per second
ACKinterval = default interval for ACK transmission of 0.01 seconds (aka SYN)
PacketSize = size of UDP packets in bits per packet

Table 16 summarizes the factors of UDT and nominates factors for further exploration based upon their ability for modification in the operational data transfer network architecture.

Table 16 - UDT Factor Summary

UDT Factor	Default Value	Nominated for Test	Representative Test Value
Synchronization Time Interval (SYN)	0.01 seconds	YES	0.01 seconds to 0.1 seconds
In-order Delivery	User selectable	NO	
UDT Maximum Segment Size (MSS)	1500 bytes	YES	256 bytes to 1400 bytes
Constant Decrease Factor (β)	1/9	NO	
Bandwidth Estimation Inter-packet Waiting Time	Every 16 packets	NO	
Target Transfer Data Rate	Unlimited	YES	Limit of 20 Mbps to 50 Mbps

Literature Review Summary

This chapter has identified existing research in the areas of network and data transfer optimization and documented how insights gained in the various topics are related to an exploration of factors that may have an effect on the response variables of the system under test. The first section defined foundational terminology in the area of network and data transfer. The second section documented areas of related research in network and data transfer optimization of data transfer network architectures with a focus on experimental design applications. This section provided key insights into areas of test and evaluation in the area of data transmission and optimization to feed decisions on appropriate factors and levels in the test design. The final section described the details of the data transfer network architecture under test and highlighted definitions of protocol design factors as they relate to the performance of the data transfer network architecture.

III. Methodology

Chapter Overview

The purpose of this chapter is to describe the process of experimental design used in this research effort and to describe the setup and execution of the test. First, the relationship of Design of Experiments to the research methodology will be described. Next, the representative test scenario to assess the relationship of input parameters to the data transfer network architecture that supports large format sensor data transmission from a forward operating location to a fixed base operating location will be described. This will include definition, documentation and rationale of the test factors, control factors, and system response variables that will be utilized in the test execution. Finally, a description of the experimental test design will be provided and the experimental procedures will be explained.

Overview of Research Methodology

Design of Experiments.

Design of experiments (DoE) applies a systematic and rigorous approach to system analysis and experimentation to ensure appropriate data is collected with requisite principles and techniques such that the data analyzed by statistical methods generates valid, defensible and supportable conclusions (Montgomery, 2009; Natrella, 2015). DoE principles may be applied to four general engineering problem areas to support comparing, characterizing, modeling and/or optimizing system performance. Comparing or screening allows one to assess whether a change in an input factor has resulted in a statistically significant change to the system response. Characterizing or ranking allows one to understand the relationship or effect of design factors as they affect the system response and provides, after design and analysis, a ranked list of important to unimportant input factors. Modeling of system performance allows one to assign a mathematical function to the system that relates input factors to system response. This relationship can then be capitalized upon to support assessment of system response through mathematical experimentation. Finally, optimizing allows one to determine the optimal settings of the input factors to support the desired system response (Natrella, 2015). This research will apply the techniques screening, ranking, modeling and assessing to support the optimization of the data transfer network architecture and application for the system under test.

Design of Experiments Terminology.

Definitions to standard terms associated with DOE are defined below (Totaro & Perkins, 2005):

Factors: The variables that affect the response variable. Factors may be classified as primary, secondary, or constant, depending on their use in an [experimental] design.
Levels: The values that a factor can assume are called its levels.
Response Variable: The measured performance of the […] system under study
Design: The experimental design specifies the number of experiments, the factor level combinations for each experiment, and the number of replications of each experiment.
Replication: […] refers to the process of repeating an experiment or set of experiments.
Main Effects: […] the main effect of a factor refers to the average change in a response variable produced by a change in the level of the factor.
Interaction Effects: Two factors interact if the performance response due to factor *i* at level *m* depends on the level of factor *j*. In other words, the relative change in the performance response due to varying factor *i* is dependent on the level of factor *j*.

The specific process for employing DoE or structured experimental design is described in a business-oriented methodology in Schmidt (2005) and more generally in Montgomery (2009) and is shown below.

Guidelines for Designing an Experiment

1. Recognition and statement of the problem
2. Selection of the response variable*
3. Choice of factors, levels, and ranges*
4. Choice of experimental design
5. Performing the experiment
6. Statistical analysis of the data
7. Conclusions and recommendations

* These steps are often performed simultaneously or in reverse order

In the first step of the DoE process, the objective of the experiment is defined to ensure a clear and accepted statement of the problem is documented which contributes to the understanding of the system being studied and the goal of the experimental solution. (Montgomery, 2009). This critical step assists in answering the intended purpose of the experiment based on the four engineering problem areas discussed previously (Natrella, 2015). This step relates to the objective of the test design and the experimental questions developed and are described in the Problem Statement and Research Objectives and Investigative Questions sections of Chapter I.

Selection of the system response variable should ensure that the chosen variable provides useful information about the system under test and aligns with the objective of the test. The response variable is generally an output of the system and must be measurable. Additionally, there may be multiple response variables measured in the experiment to allow for assessment of performance from varying perspectives (Montgomery, 2009). The rationale for selection of the system response variables based on related research is documented in Chapter II while the definition of the system response variables is documented later in this chapter.

Design factors are those which may influence the response of the system under test. When choosing the experimental factors it is important to rely on practical experience and theoretical understanding of the system under test to ensure an appropriate design region is chosen for each variable (Montgomery, 2009). It is also important to not approach an experiment with previous bias as that may skew results (Montgomery, 2009). The rationale for selection of the system response variables based on related research and understanding of the operational data transfer network architecture is documented in Chapter II while the definition of the system response variable is documented later in this chapter.

Choice of the experimental design should consider the experimental objectives (Montgomery, 2009). By planning the experiment, it is generally assumed that "some of the factor levels will result in different values for the response. Consequently, [the experimenter] is interested in identifying which factors cause this difference and in estimating the magnitude of the response change" (Montgomery, 2009). The experimental design can also be affected by the amount of resources available for the experiment (Schmidt, 2005).

In performing the experiment it is imperative to carefully monitor progress to ensure the execution is performed as planned. Errors in experimental execution will usually remove any validity (Montgomery, 2009).

If the design and execution of the experiment has been performed correctly then the statistical analysis of the collected data can be performed in a methodical manner that leads to objective results (Montgomery, 2009). Development of an empirical model that expresses the relationship between the design factors of interest and the system response within the experimental range allows for keen insights into expected system performance under a range of input parameters.

Finally, once the analysis of the data is complete, practical conclusions about the results and a recommended course of action can be created. Iterations may need to be performed on specific areas within the experiment to support validation of the experimental results as they relate to the system under test (Montgomery, 2009).

Detailed System Description

The high-level definition of the operational data transfer network architecture is documented in Chapter II. This section defines the details of the system under test and establishes the precedence for the experimental design. To support the definition, a Process Flow Diagram of the data transfer, as shown in Figure 7, was created to document the data transfer portion of the operational data transfer network. The documented process flow represents how data is passed between the forward data interface, the sender, and the archival data interface, the receiver, with the UDT data transfer application as represented in Figure 6 of Chapter II.

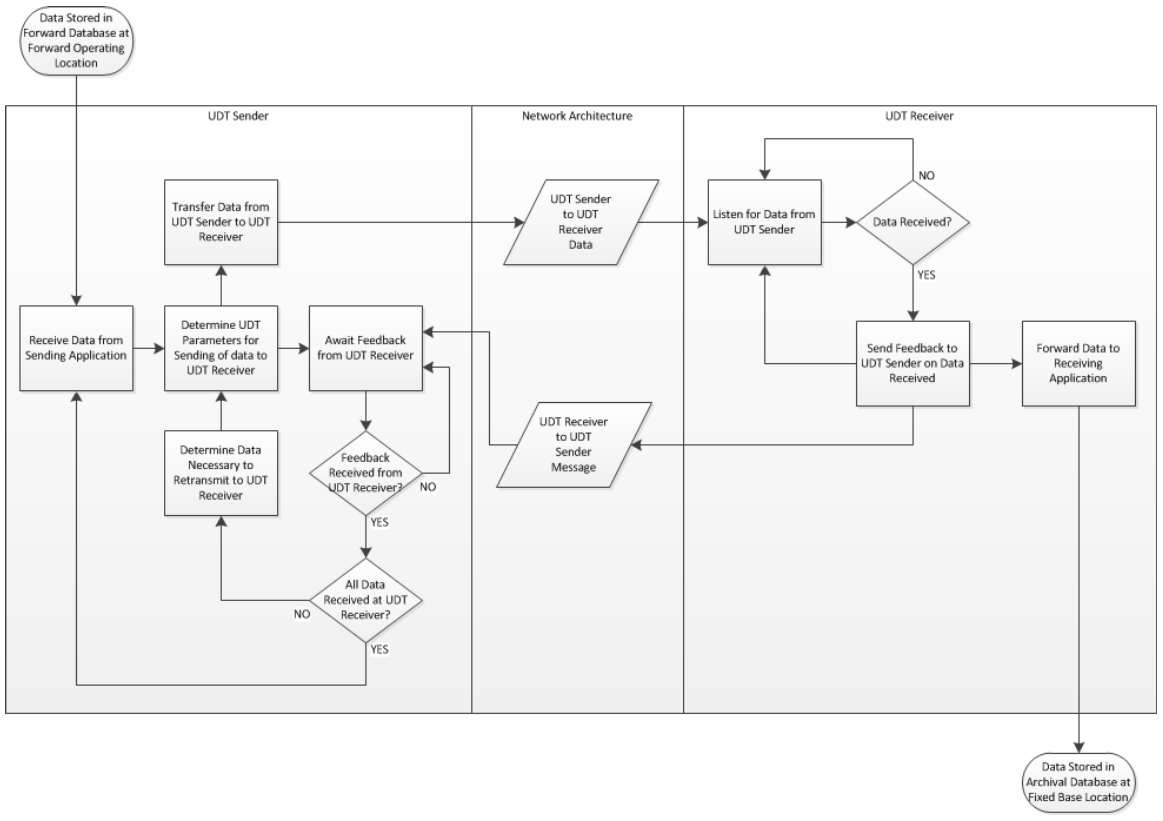

Data Stored in Forward Database at Forward Operating Location
UDT Sender
Network Architecture
UDT Receiver
Transfer Data from UDT Sender to UDT Receiver
UDT Sender to UDT Receiver Data
Listen for Data from UDT Sender
Data Received?
NO
YES
Receive Data from Sending Application
Determine UDT Parameters for Sending of data to UDT Receiver
Await Feedback from UDT Receiver
Send Feedback to UDT Sender on Data Received
Forward Data to Receiving Application
UDT Receiver to UDT Sender Message
Determine Data Necessary to Retransmit to UDT Receiver
Feedback Received from UDT Receiver?
NO
YES
All Data Received at UDT Receiver?
NO
YES
Data Stored in Archival Database at Fixed Base Location

Figure 7 - Process Flow Diagram of Data Transfer Architecture

To support further test design activities, the process flow diagram was used to build a Cause-and-Effect diagram that depicts possible causes (process inputs or factors) for a response by the system (response variables). The Cause-and-Effect (CE) diagram highlights inputs to the system and their relationship to the planned test design. These are highlighted on the diagram with the markings of (C), (N) and (X), where (C) depicts inputs that can be held constant, (N) represents inputs that are not controllable and therefore represent noise in the system, and (X) represents the inputs (test design factors) that are the focus of the experimental process. The CE diagram for the data transfer network architecture is depicted in Figure 8.

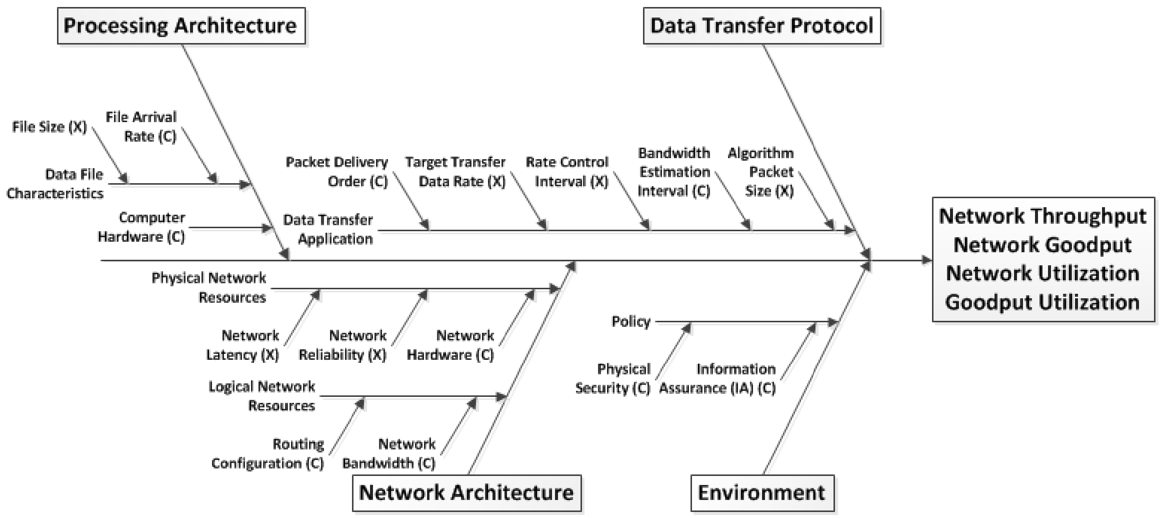


Figure 8 - Cause-and-Effect Diagram for System Response

Description of System Variables

Based on the output of the results of the CE diagram, Figure 9 describes depicts the system under test in an input, process, output (IPO) diagram. In the IPO diagram, the experimental inputs to the system are shown entering the process from the left and are the factors for the DoE. These variables will be controlled during experimentation and will vary based on the setting described later in this section. The outputs of the process are measurable response variables that represent the experimental results utilized to evaluate the performance of the system. The inputs entering the process from below the system are environmental and experimental variables that are not variable and will be held constant during the experimental process.

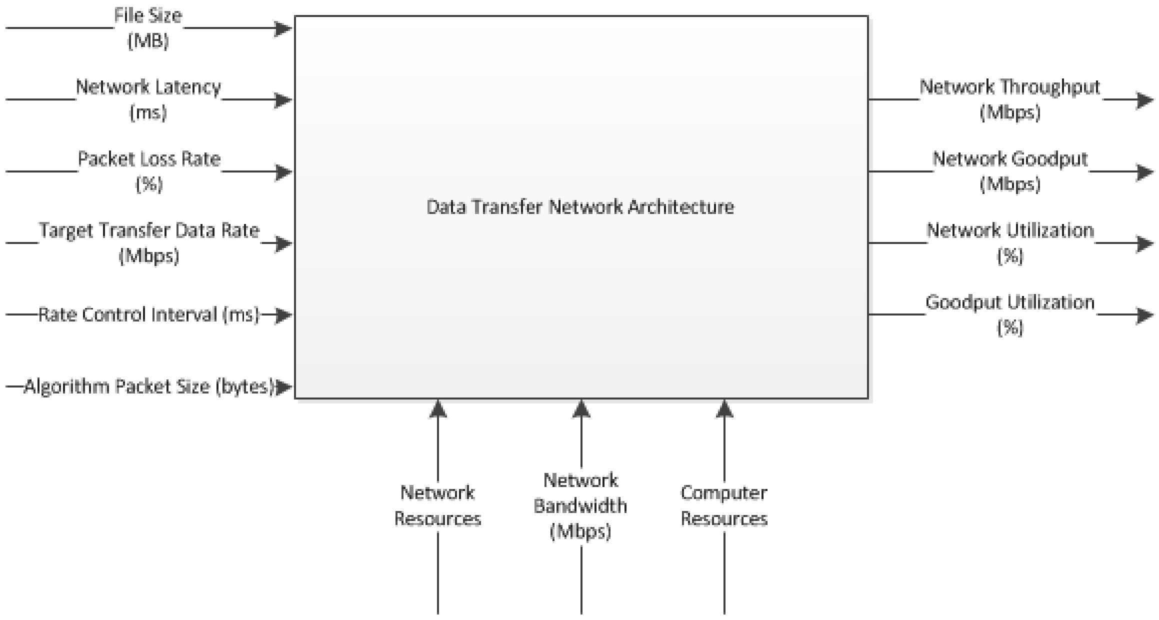


Figure 9 - Input, Process, Output (IPO) Diagram

Control Variables.

Network Bandwidth: This control variable represents the allocated transmission rate of the network architecture in megabits per second (Mbps). The value of this variable will be held constant based on the limit set on the operational network. From the literature review, test points often spanned ranges from 1 Mbps to 10 Gbps but were generally limited to comparisons on networks operating at speeds under 100 Mbps. This control variable will be held constant at 50 Mbps.

Network Resources: As described in the Assumptions and Limitations section of Chapter I, the configuration of the network resources in the operational network are governed by an external entity and are expected to meet performance requirements levied by the customer. The requirement for a Layer 3 virtual private network (VPN) with a bandwidth of 50 Mbps drives the implementation of the network resources between sender and receiver location points of presence (PoP). It is assumed that the underlying network architecture of the physical layer/Layer 1, link layer/Layer 2 and network layer/Layer 3 will remain constant. The intent of this control ensures a constant bandwidth, network induced latency and static routing profiles during test. The specification of end-to-end network latency will be varied, however it will be held static during a test run to a minimum value. Due to this control, any additional latency observed within the network is assumed to be a result of over-utilization by the data transfer application through excessive network congestion.

Application Protocol: UDT will be used in its UDP file transfer mode to represent configuration of operational network.

End-Point Computing Resources: As described in the Assumptions and Limitations section of Chapter I, the hardware configuration of the end-point sender and receiver (e.g. processors, memory, and network interface card) will remain static during the testing as it is assumed that they do not limit data transmission when compared to the data transfer network architecture.

Response Variables.

Network Throughput: The raw amount of data transferred between sender and receiver measured over time. This response variable includes packet overhead and data retransmissions. The use of overall throughput will allow one to determine if overall data transmission is overwhelming the network architecture. The units for this variable are megabits per second (Mbps) where 1,000,000 bits per second equals 1 Mbps.

Network Goodput: The specific amount of data in packet payloads transferred between the sender and receiver measured over time. This metric does not include information included in protocol overhead or retransmission of data due to network congestion or network loss and is referred to as, “ the effective amount of data delivered through the network” (Xu, Tian, & Ansari, 2004) or “the number of unique packets delivered to an end host in a given amount of time, as opposed to the total number of packets transmitted in a given amount of time that includes retransmissions” (Sharma, Gillies, & Feng, 2010). By definition, the measure of *Network Goodput* is always less than or equal to *Network Throughput*.

Network Utilization: The relationship of the amount of network resources being utilized for the transmission of data as compared to the *Target Transfer Data Rate* factor. The target transfer rate of the data transfer algorithm can be set by the user to allow for managed control of the available network resources. This user control allows for control of the aggressiveness of the data transfer algorithm through this limiting factor. By taking a more aggressive transfer posture the user would set the target transfer rate to consume all available bandwidth as related to the *Network Bandwidth* factor, whereas, a less aggressive transfer posture would be set at a rate less than the expected *Network Bandwidth* factor. This response variable will be normalized into percentage of utilization targeted by the transfer algorithm. Although the *Network Bandwidth* will not vary, the *Target Transfer Data Rate* will be variable, thus the *Network Utilization* must be based on the *Target Transfer Data Rate* instead of the usual factor of *Network Bandwidth*. This variable is defined in Equation 13.

$$Network\ Utilization = \frac{Network\ Throughput}{Target\ Transfer\ Data\ Rate} \times 100$$

Equation 13 - Network Utilization

Where:
Network Throughput = the raw amount of data transferred between sender and receiver over time
Target Transfer Data Rate = the factor that represents the targeted transmission rate of the network transfer algorithm

Goodput Utilization: The relationship of the amount of network resources being utilized for the transmission of only data associated with *Network Goodput* as compared to the *Target Transfer Data Rate* factor. This response will be normalized into percentage of utilization targeted by the transfer algorithm. Although the *Network Bandwidth* will not vary, the *Target Transfer Data Rate* will be variable, thus the *Goodput Utilization* must be based on the *Target Transfer Data Rate* instead of the usual factor of *Network Bandwidth*. This variable is defined in Equation 13.

$$Goodput\ Utilization = \frac{Network\ Goodput}{Target\ Transfer\ Data\ Rate} \times 100$$

Equation 14 - Network Utilization

Where:
Network Goodput = the specific amount of data in packet payloads transferred between the sender and receiver measured over time
Target Transfer Data Rate = the factor that represents the targeted transmission rate of the network transfer algorithm

Experimental Factors.

Network Latency: This factor represents the end-to-end round-trip time (RTT) of the data transfer network that connects the sending and receiving nodes. Observations of the operational network show variation of this factor between values of 200 ms and 500 ms. From the definition of the bandwidth delay product as described on page 20 of Chapter II, it is expected that an interaction effect between the target transfer data rate and network latency may be observed. In order to capture the performance of the system under test, this factor will be set to the values shown in Table 17. From the review of related research where latencies were varied between 0 ms and 550 ms, the delta of 300 ms between high and low factors it is expected that these settings should be large enough to enable a respective change in the response variables.

Table 17 - Network Latency Factor Settings

Factor	**Discrete Settings**	
	High (+)	**Low (-)**
Network Latency	500 ms	200 ms

Packet Loss Rate: Due to the results of the literature, the packet loss rate will be varied to allow for discovery of any relationship between the factor and the response variables. This loss is not due to congestion but due to network transmission errors. Test parameters in literature varied this factor between 0% and 10% loss. In an attempt to capture the performance of the operational network, this factor will be set to the values shown in Table 18.

Table 18 - Packet Loss Rate Factor Settings

Factor	**Discrete Settings**	
	High (+)	**Low (-)**
Packet Loss Rate	5%	0%

File Size: The size of the files transferred in the specific operational network, as defined in Chapter II, varies between files with sizes in the tens of kilobytes range, to files with sizes in the single gigabytes range. The majority of the data transferred via the operational network lies in the tens of megabytes to hundreds of megabytes range. It is for this purpose that the file size used in testing will be varied based on the file size of the majority of data that is transferred via the operational network to allow for any observed interactions in data transmission performance and the transferred file size. In an attempt to characterize the performance of the operational network during its most used states, this factor will be set to the values shown in Table 19.

Table 19 - File Size Factor Settings

Factor	**Discrete Settings**	
	High (+)	**Low (-)**
File Size	300 MB	5 MB

Target Transfer Data Rate: The UDT application protocol allows for insertion of a target transfer data rate. If this factor is set to unlimited, it allows the protocol to attempt to use all available transfer bandwidth provided by the data transfer network architecture. The provisioning for the bandwidth of the operational data transfer network architecture is set to allow for 50 Mbps transfer rate, however, based on observed performance throughput rarely achieves a rate near to this resource setting. This factor represents the configuration setting within the sending application that will be used by UDT for the targeted transfer speed of the data. This factor will be based on the environment variable *Network Bandwidth* to ensure the maximum targeted transfer rate does not exceed the provisioned data rate. In order to observe any interactions with this factor and the other input factors, this factor will be set to the values shown in

Table 20.

Table 20 - Target Transfer Data Rate Factor Settings

Factor	Discrete Settings	
	High (+)	**Low (-)**
Target Transfer Data Rate	50 Mbps	10 Mbps

Rate Control Interval: The rate control of the protocol is governed by the Synchronization time interval, or SYN. By default this factor is set to 0.01 seconds, however this variable in configurable. As was seen previously in UDP-based Data Transfer Protocol section in Chapter II, describing test methodologies, the test networks under consideration had considerably less resource constraints restraints than the operational network in question (e.g. higher network bandwidth and lower network latency). Due to this observation, the impact of this factor under operational network constraints is of interest. Feedback received per SYN interval on the higher bandwidth networks involves on the order of twenty times the amount of packets delivered in the same period as with the operational network. In order to observe these potential effects, this factor will be set to the values show in Table 21.

Table 21 - Rate Control Interval Factor Settings

Factor	Discrete Settings	
	High (+)	**Low (-)**
Rate Control Interval	0.1 sec	Default (0.01 sec)

Algorithm Packet Size: The UDT application protocol allows for the configuration of the maximum segment size of the packets transmitted via UDT. The protocol documentation indicates that, “In most situations, the optimal UDT packet size is the network MTU [maximum transmission unit] size” with a default value of 1500 bytes (Gu, 2011). Changes in the packet size that the algorithm utilizes affect the overhead of the network. For example, with the default value of 1500 bytes and taking into account the UDT data packet header structure that consists of 12 bytes of information the payload of the UDT data packet contains 1488 bytes whereas with a programmed value of 512 bytes, the payload of the UDT data packet contains only 500 bytes (Gu & Grossman, 2007).

The tradeoff between these values becomes intermingled with the probability of error while the data packet is transmitted versus the additional overhead required to send the same amount of data. Observations from the use of the operational network revealed fragmentation of packets with sizes greater than 1400 bytes while related research varied data packet size between 400 and 1500 bytes, as shown in Table 14. In order to observe these potential effects and maintain a representative input factor for use within the operational network, this factor will be set to the values shown in Table 22.

Table 22 - Algorithm Packet Size Factor Settings

Factor	**Discrete Settings**	
	High (+)	**Low (-)**
Algorithm Packet Size	1400 bytes	256 bytes

Experimental Design

Experimental Equipment.

In order to execute the test, a laboratory network will be configured as depicted in Figure 10.

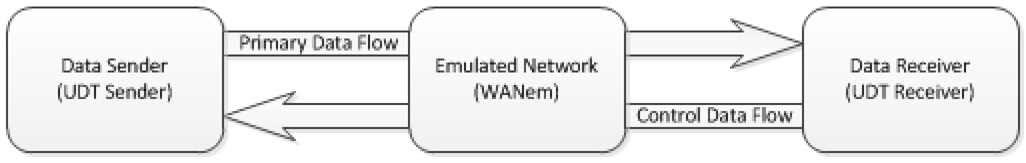
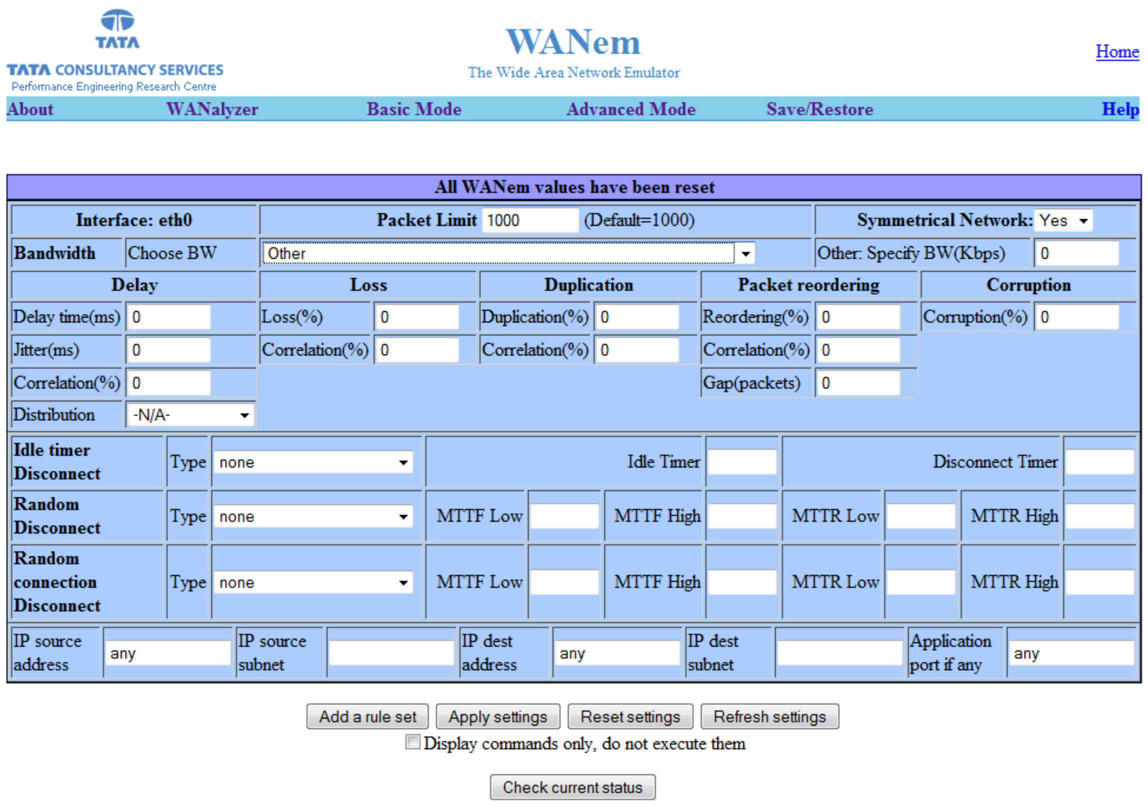


Figure 10 - Network Experimental Setup

Due to the administrative limitations of the operational network which does not allow for control of the network performance factors the network performance will be emulated. The network emulation tool chosen for this experiment is WANem.

Network Emulation.

WANem is a research and development network emulation tool developed by the Performance Engineering Research Centre, TATA Consultancy Services (PERC - TATA Consultancy Services, Ltd., 2014). The tool is, "meant to provide a real experience of a Wide Area Network/Internet, during application development/testing over a LAN environment" (PERC - TATA Consultancy Services, Ltd., 2014). The WANem interface is shown in Figure 11.


TATA
TATA CONSULTANCY SERVICES
Performance Engineering Research Centre
WANem
The Wide Area Network Emulator
Home
About
WANalyzer
Basic Mode
Advanced Mode
Save/Restore
Help

All WANem values have been reset

Interface: eth0		Packet Limit	1000	(Default=1000)	Symmetrical Network:	Yes	
Bandwidth	Choose BW	Other			Other: Specify BW(Kbps)	0	

Delay		Loss		Duplication		Packet reordering		Corruption	
Delay time(ms)	0	Loss(%)	0	Duplication(%)	0	Reordering(%)	0	Corruption(%)	0
Jitter(ms)	0	Correlation(%)	0	Correlation(%)	0	Correlation(%)	0		
Correlation(%)	0					Gap(packets)	0		
Distribution	-N/A-								

Idle timer Disconnect	Type	none	Idle Timer				Disconnect Timer			
Random Disconnect	Type	none	MTTF Low		MTTF High		MTTR Low		MTTR High	
Random connection Disconnect	Type	none	MTTF Low		MTTF High		MTTR Low		MTTR High	

IP source address	any	IP source subnet		IP dest address	any	IP dest subnet		Application port if any	any		

Figure 11 - WANem Advanced Mode Interface

Factors that will affect network configuration settings will be programmed into WANem to allow for variation of the factors as directed by the test design. The setting of Bandwidth (Other) with a specified rate as in the planned test design will set the control input factor of *Network Bandwidth.* The WANem settings of Delay time (ms) and Loss (%) will be used to control the variable input factors of *Network Latency* and *Network Reliability* based upon the settings in the planned test design. All other settings available via the WANem interface will be set to their default value of zero, thus are not included in the network emulation.

Data Transfer Mechanism.

The UDT data transfer protocol was configured via user definable parameters on the UDT sender and UDT receiver for the variable factors of *Target Transfer Data Rate*, *Rate Control Interval* and *Algorithm Packet Size* while the control factors of the algorithm of *Packet Delivery Order* and *Bandwidth Estimation Interval* were held constant. *Packet Delivery Order* was set to allow for out-of-order delivery of packets and *Bandwidth Estimation Interval* was set at the protocol default of 16 packets.

The factor *File Size* was controlled by only transmitting data files of the size indicated in the test design. By sending only files of the size indicated in the test design to the UDT sender, the files transferred via the emulated network represented those dictated by the test design.

System response variables were measured via reported statistics from the UDT application through the use of UDT performance monitor (perfmon) and summary statistics gathered from the UDT sender and UDT receiver through the use of accumulators that reported the data sent from the UDT sender and data received at the UDT receiver. Details of the data available from perfmon via the traceinfo UDT structure are listed in Appendix B. The data collected was used to calculate the system response variables described previously.

Experimental Procedures.

The experimental setup depicted in Figure 12 was used in a network laboratory to allow for collection of the data transfer network architecture response. By routing all traffic between the UDT Sender and the UDT Receiver through the WANem network interface, the network emulation factors of *Network Bandwidth*, *Network Latency* and *Network Reliability* will be applied.

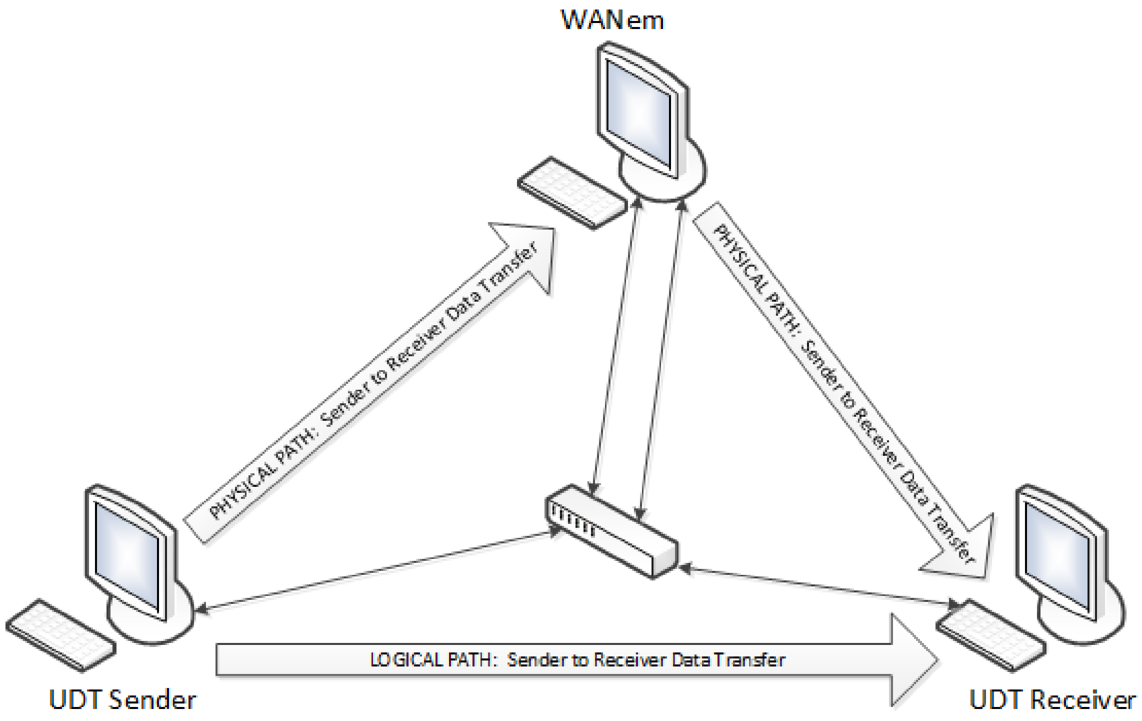

WANem
PHYSICAL PATH: Sender to Receiver Data Transfer
PHYSICAL PATH: Sender to Receiver Data Transfer
LOGICAL PATH: Sender to Receiver Data Transfer
UDT Sender
UDT Receiver

Figure 12 - Experimental Setup

For each experimental run, the factors will be configured into the network emulator, the data transfer protocol and the input file configuration. The data transfer was initiated via execution of the data transfer protocol send and was allowed to transfer data for 120 seconds while beginning a new file transfer if the simulation time had not reached this time prior to the next file transfer being initiated. This allows for sufficient time for data to complete at least a single file transfer from the sender to the receiver at the varying theoretical data rates and file sizes. Upon the completion of the latest initiated file transfer, the data transfer was discontinued and the response statistics gathered for calculation for the current experimental run. The experimental runs were generated and executed based upon randomized test designs created using JMP.

Screening Design Pilot Study.

A pilot study was used to screen for the "[separation of] the vital few [factors] from the trivial many [factors]" (Schmidt, 2005). Two screening designs were developed to support the verification of the discrete factor settings and to validate the collection of the data used for the calculation of the experimental response variables. As the pilot study was primarily concerned with validating data collection and the factor main effects to validate input settings, confounding of the main and two-way interaction effects was acceptable. The system response of interest in these designs focuses on the response variable of *Goodput Utilization*. This response variable represents the normalized utilization of the data transfer network architecture for actual data file transfer use and decouples the limitation of transfer speed from the *Target Transfer Data Rate* as it relates to system performance.

To support the pilot study, two independent tests were performed. The first utilized a twelve-run Plackett-Burman design, as recommended by Schmidt (2005). This design was created using JMP and executed in the laboratory testbed. This design is captured in Table 23.

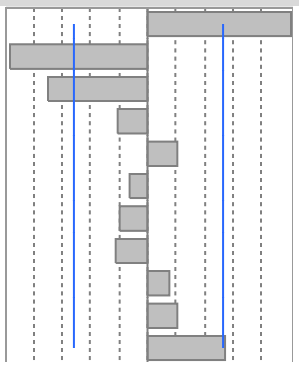
Table 23 - Plackett-Burman 12-Run Screening Design

Run	File Size (MB)	Network Latency (ms)	Packet Loss Rate (%)	Target Transfer Data Rate (Mbps)	Rate Control Interval (sec)	Algorithm Packet Size (bytes)
1	5	200	5	50	0.01	1400
2	300	500	0	50	0.01	1400
3	5	200	0	10	0.01	256
4	300	200	0	50	0.1	1400
5	5	500	5	50	0.1	256
6	300	200	5	10	0.01	256
7	300	500	0	10	0.1	256
8	300	200	5	50	0.1	256
9	5	500	5	10	0.1	1400
10	5	200	0	10	0.1	1400
11	5	500	0	50	0.01	256
12	300	500	5	10	0.01	1400

Representative results from the Plackett-Burman screening design are shown in Table 24 with raw results documented in Appendix D - Raw Experimental Data and system response variables in Appendix E - Calculated System Response Variables.

Table 24 - Contrasts from Plackett-Burman Screening Design

Term	Contrast	Individual p-Value
Algorithm Packet Size	17.4701	0.0182*
Target Transfer Data Rate	-16.9134	0.0193*
Network Latency	-12.3039	0.0426*
Rate Control Interval	-3.6320	0.5002
File Size	3.5817	0.5548
Packet Loss Rate	-2.2992	0.7045
Algorithm Packet Size*Target Transfer Data Rate	-3.4541 *	0.5708
Algorithm Packet Size*Network Latency	-3.8300 *	0.4442
Target Transfer Data Rate*Network Latency	2.6768 *	0.6578
Algorithm Packet Size*Rate Control Interval	3.7270 *	0.4567
Target Transfer Data Rate*Rate Control Interval	9.5208 *	0.0898



Lenth PSE = 4.952 (blue lines)
Null-space terms were added after factor space was exhausted.
Asterisked terms were forced orthogonal. Analysis is order dependent.
P-Values derived from a simulation of 10000 Lenth t ratios.

Through examination of the contrasts from the Plackett-Burman Screening Design for the system response variable of Goodput Utilization, the input factors of *Algorithm Packet Size*, *Target Transfer Data Rate*, *Network Latency* are significant at the p = 0.05 level indicating that the factors would significantly impact the model in subsequent runs and should be examined in further test designs. This finding is also apparent in the Half Normal Plot shown in Figure 13 where factors that are not significant fall on or near the linear collection response values (Natrella, 2015) and the input factors of *Algorithm Packet Size*, *Target Transfer Data Rate*, *Network Latency* diverge from the linear plot.

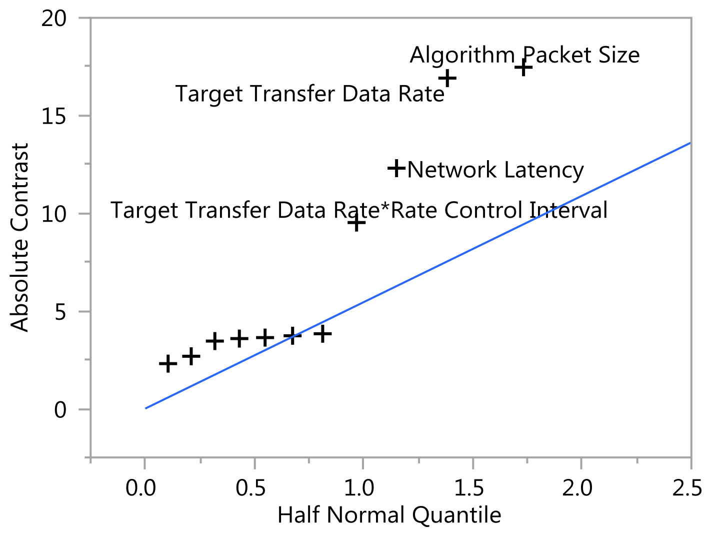
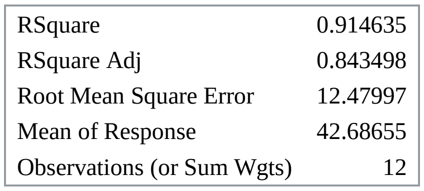


Figure 13 - Half Normal Plot for Plackett-Burman Screening Design

The interaction effect of *Target Transfer Data Rate * Rate Control Interval* diverges from the linear plot in Figure 13. As this interaction is of interest based on the link to the sampling rate of the data transfer algorithm and due to the fact that this effect is significant at the p = 0.1 level, this factor was left in the model. In order to maintain hierarchy, the input factor of *Rate Control Interval* was added for model consideration due to the significance of the interaction effect of *Target Transfer Data Rate * Rate Control Interval*.

Although a model of system performance would generally not be created from a screening design as one is attempting to estimate only main factor effects, a model was created in JMP to provide confidence in the screening results. The summary of fit for the Plackett-Burman screening design model based upon the main effects from the screened input factors of *Algorithm Packet Size*, *Target Transfer Data Rate*, *Network Latency* and *Rate Control Interval*, (to maintain hierarchy) and the interaction effect of *Target Transfer Data Rate * Rate Control Interval* is shown in Table 25.

Table 25 - Plackett-Burman Summary of Fit



RSquare	0.914635
RSquare Adj	0.843498
Root Mean Square Error	12.47997
Mean of Response	42.68655
Observations (or Sum Wgts)	12

Due to the intent of data collection for the pilot study, only one replication was performed in the design; however the model exhibited high fit with an adjusted R-squared value of 0.84 that indicates that a high amount of the variation in the data can be attributed to the model vice residual error. The fact that the R-squared value of 0.91, which includes all input factor effects, and the adjusted R squared value, which includes only those deemed significant, differ by only 0.07, allows for confidence that the significant effects of *Algorithm Packet Size*, *Target Transfer Data Rate*, *Network Latency* and *Rate Control Interval* and *Target Transfer Data Rate * Rate Control Interval* account for the variance in *Goodput Utilization*. The Actual by Predicted plot, shown in Figure 14, graphically demonstrates the fit of the model with the actual results appearing along the predicted response.

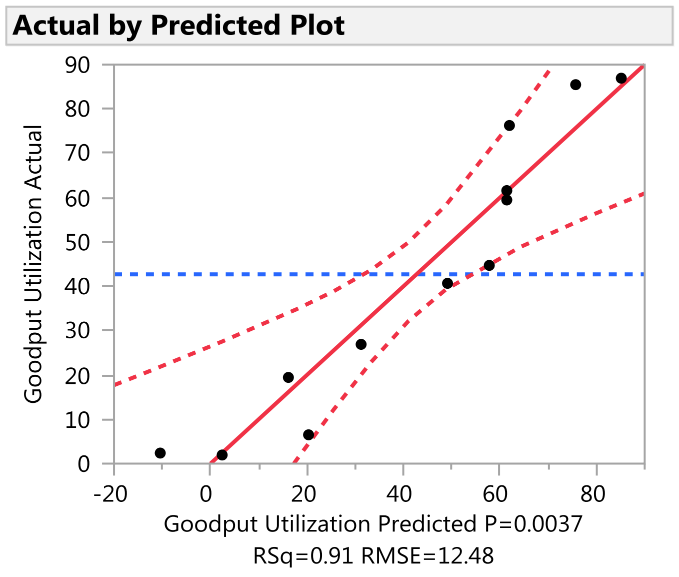


Figure 14 - Plackett-Burman Screening Design Actual by Predicted Plot

As the Plackett-Burman design can alias main effects with several two-way interaction effects (Schmidt, 2005; SAS Institute Inc., 2015) and potentially confound two-way interaction effects a Definitive Screening Design was utilized to test for potential two-way interaction effects and curvature of the response in the design. To estimate non-linear effects with this design required insertion of center points to the design runs that inform of non-linear effects rather than identifying the responsible factors (SAS Institute Inc., 2015). A Definitive Screening Design was created in JMP and executed in the laboratory test bed. The design is captured in Table 26.

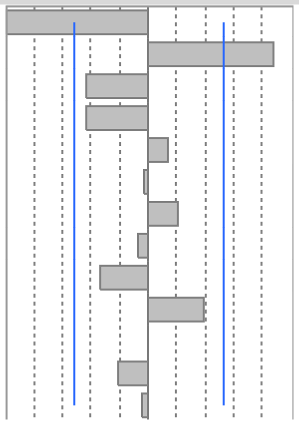
Table 26 - Screening Design Run Matrix

Run	File Size (MB)	Network Latency (ms)	Packet Loss Rate (%)	Target Transfer Data Rate (Mbps)	Rate Control Interval (sec)	Algorithm Packet Size (bytes)
1	300	200	0.5	20	0.1	1500
2	5	200	0	50	0.055	1500
3	300	500	1	20	0.055	256
4	152.5	350	0.5	35	0.055	878
5	152.5	500	1	50	0.1	1500
6	152.5	200	0	20	0.01	256
7	5	500	0.5	50	0.01	256
8	5	350	1	20	0.01	1500
9	300	200	1	50	0.01	878
10	5	200	1	35	0.1	256
11	300	500	0	35	0.01	1500
12	152.5	350	0.5	35	0.055	878
13	300	350	0	50	0.1	256
14	5	500	0	20	0.1	878

Representative results generated from JMP via Screening Analysis from the Definitive Screening Design are shown in Table 27 with raw results documented in Table 44 of Appendix D - Raw Experimental Data and system response variables in Table 48 of Appendix E - Calculated System Response Variables.

Table 27 - Contrasts from Definitive Screening Design

Term	Contrast	Individual p-Value
Target Transfer Data Rate	-16.2248	0.0162*
Algorithm Packet Size	14.2864	0.0225*
Packet Loss Rate	-6.9998	0.1528
Network Latency	-6.9729	0.1545
File Size	2.2195	0.6785
Rate Control Interval	-0.4739	0.9295
Target Transfer Data Rate*Target Transfer Data Rate	3.4105	0.4576
Target Transfer Data Rate*Algorithm Packet Size	-1.1421	0.8318
Algorithm Packet Size*Algorithm Packet Size	-5.3697 *	0.2546
Target Transfer Data Rate*Packet Loss Rate	6.3326 *	0.1898
Algorithm Packet Size*Packet Loss Rate	-0.0513 *	0.9931
Packet Loss Rate*Packet Loss Rate	-3.3013 *	0.5002
Null14	-0.5945	0.9097



Lenth PSE = 4.952 (blue lines)
Null-space terms were added after factor space was exhausted.
Asterisked terms were forced orthogonal. Analysis is order dependent.
P-Values derived from a simulation of 10000 Lenth t ratios.

Through examination of the contrasts from the Definitive Screening Design for the system response variable of Goodput Utilization, the input factors of *Target Transfer Data Rate* and *Algorithm Packet Size* are significant at the p = 0.05 level and should be examined in further test designs. This finding is also apparent in the Half Normal Plot shown in Figure 15 where factors that are not significant at the 0.05-level fall on or near the linear collection response values (Natrella, 2015) and the input factors of *Target Transfer Data Rate* and *Algorithm Packet Size* which are significant diverge from the linear plot.

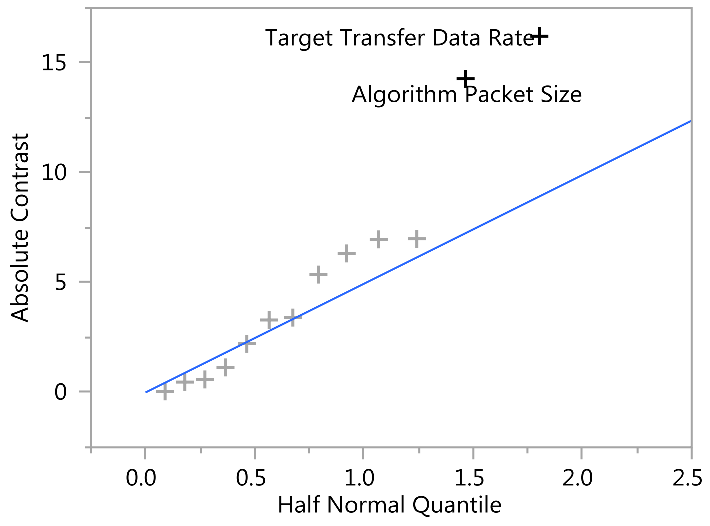


Figure 15 - Half Normal Plot for Definitive Screening Design

Again, although a model of system performance would generally not be created from a screening design as one is attempting to estimate only main factor effects, a model was created in JMP to provide confidence in the screening results. The summary of fit for the Definitive Screening Design model based upon the main effects from the screened input factors of *Target Transfer Data Rate* and *Algorithm Packet Size* is shown in Table 28.

Table 28 - Definitive Screening Summary of Fit

RSquare	0.704638
RSquare Adj	0.650936
Root Mean Square Error	15.78996
Mean of Response	38.36506
Observations (or Sum Wgts)	14

The R-squared adjusted value of 0.65, although lower than the Plackett-Burman screening design produced, still indicates that the design captures the majority of the response from the system and suffices for use in a screening design. This change in value is likely attributed to the relatively large contrast values for the *Packet Loss Rate* and *Network Latency* input factors as compared to the results from the previous screening design. The fact that the R-squared value of 0.70, which includes all input factor effects, and the adjusted R squared value, which includes only those deemed significant, differ by only 0.05, allows for confidence that the significant effects of *Target Transfer Data Rate* and *Algorithm Packet Size* account for the variance in *Goodput Utilization*

The inclusion of center points in the design allowed for the ability to test for a non-linear response. The distribution of the model fit can be seen in the Actual by Predicted plot, shown in Figure 16, where for screening purposes, the actual response is generally predicted by the model and does not indicate the presence of curvature as the values are evening distributed along the linear estimation.

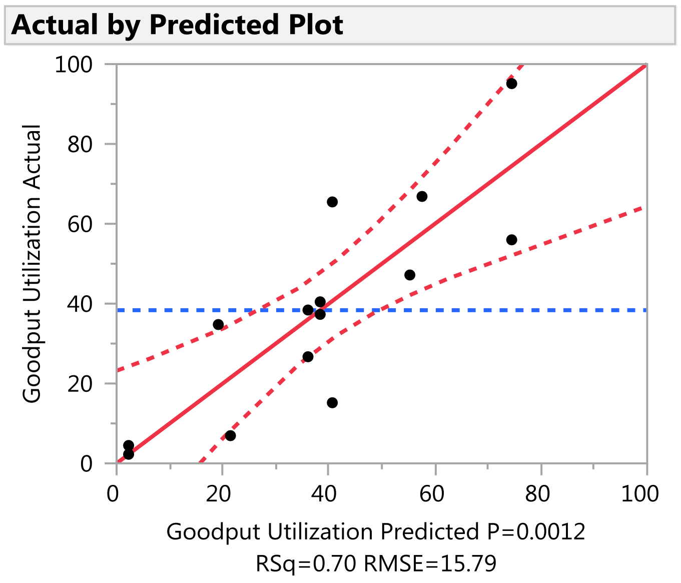


Figure 16 - Definitive Screening Design Actual by Predicted Plot

The creation of a screening design and pilot study served two purposes. The first purpose ensured the ability to collect the appropriate system responses to enable calculation of the network architecture under test response variables. The second purpose allowed for an initial regression step for the system input factors that were not significant to the system response. Based on the results of the screening design, the input factors of interest for future testing are able to be limited to the *Algorithm Packet Size*, *Target Transfer Data Rate, and Network Latency* and, to maintain hierarchy, the *Rate Control Interval*. The factors that can be fixed within the design space and therefore removed from further experimentation were *Transfer File Size* and *Packet Loss Rate* as their p-values indicated from the screening designs indicated that the effect of these input factors were not significant at the 0.05-level. The indicated adjusted R-squared values of 0.8435 for the Plackett-Burman screening design and 0.6509 for the Definitive Screening Design based indicated the designs account for a majority of the variation in the system response for *Goodput Utilization*.

Test Design.

Prior testing of the data transfer network architecture has been performed via a one- factor-at-a-time (OFAT) test methodology. This process established a baseline setting of levels for each factor and then methodically varied each individual factor while capturing system response variables. The major disadvantage of this test methodology is that possible interactions between factors are not considered. An interaction is defined as “the failure of one factor to produce the same effect on the response at different levels of another factor” (Montgomery, 2009). Thus, accurate assessments of the UDT protocol performance on the operational data transfer network architecture have only been observed as the effect of a single factor without regard to the potential relationship of the system performance to other factors. In order to assess the relationship of the factors and their interactions on the performance of the data transfer network architecture, a full factorial experiment was chosen based on the number of input factors and expected interactions.

Due to the results of the Pilot Study, a new IPO diagram is required to document the system input factors for test is shown in Figure 17. The new diagram highlights the shift of the input factors of *Network Reliability* and *File Size* from ones that vary in the test design to control factors.

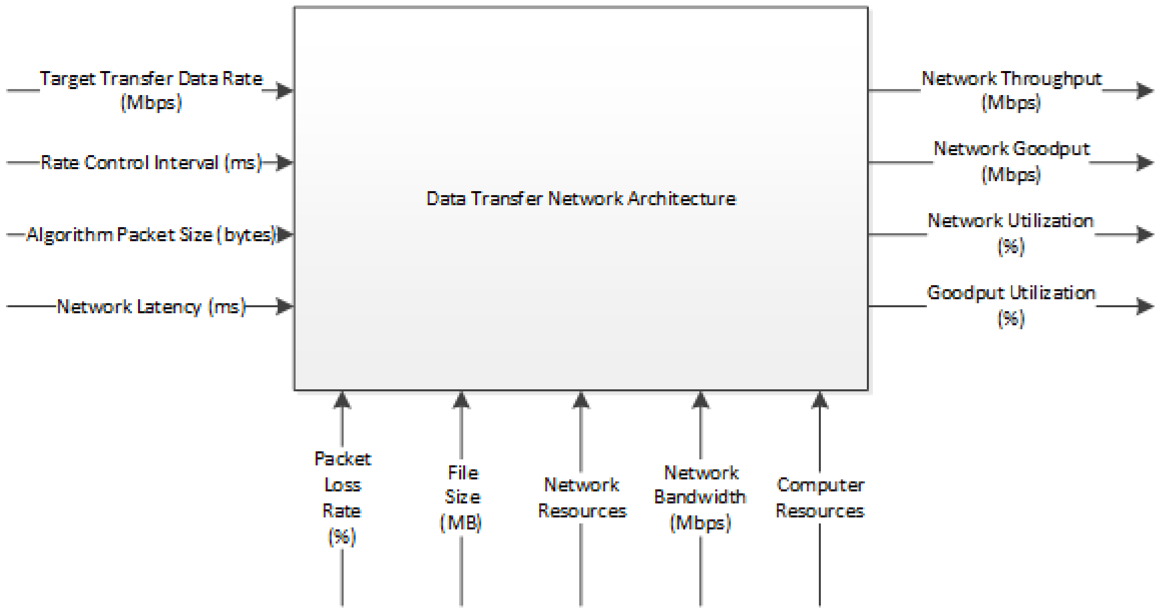


Figure 17 - IPO Diagram for Full Factorial Design

As the screening results drove toward an experimental setup with 4 input factors, all with two-levels, Schmidt (2005) recommends utilization of a full factorial design with greater-than-or-equal to three repetitions to allow for 95% confidence in the estimate of predicted standard deviation and 99.99% confidence in the estimate of predicted response. Schmidt's design selection process is depicted in Figure 21 of Appendix A.

Based on the results of the screening design in the pilot study, the factors affecting the system under test are well understood and relate to relevant research. They capture the operational network performance parameters so as to allow a direct translation from the emulative network test environment to the operational network environment. The use of a factorial design is preferred to allow for full modeling of interaction effects within the system. Had the original set of six input factors been required, (Schmidt, 2005) recommends use of a fractional factorial design, in this case a 16-run fractional factorial as depicted in Figure 22, however, this design is a Resolution IV design and would result in aliasing of 2-way interaction effects which is not desired for this effort. A 2^4 full factorial design will be utilized for this experiment. As there are sixteen possible combinations in a 2^4 design, three replications will be utilized thus comprising the test design of 64 test runs. A test matrix was created in JMP for the 2^4 factorial design with three replications and with a randomized run order. The run matrix is documented in Table 41 through Table 42 of Appendix C - Test Design.

The input factor settings for the full factorial design are depicted in Table 29 with the Fixed Settings chosen to fall within the bounds of the screening results and representative of the operational data transfer network architecture. One important item to note is the change in value of the control factor of *Network Bandwidth*. During the Pilot Study, it was discovered that the operational network was only operating under an allocation of 40 Mbps bandwidth vice the previously understood value of 50 Mbps. The justification for this change is beyond the scope of this effort. In order to maintain representative results from the emulated network, the control factor value was reduced to 40 Mbps. Due to the validation of a linear response for *Goodput Utilization*, this change does not impact the validity of the results presented.

Table 29 - Full Factorial Test Design Factor Settings

Factor	**Discrete Settings**	
	High (+)	**Low (-)**
Target Transfer Data Rate	40 Mbps	10 Mbps
Algorithm Packet Size	1400 bytes	256 bytes
Rate Control Interval	0.1 ms	0.01 ms
Network Latency	500 ms	200 ms
	Fixed Settings	
Network Bandwidth	40 Mbps	
File Size	200 MB	
Packet Loss Rate	2.5%	

Summary

This chapter described the process of design of experiments (DoE) used in this research effort and explained the setup and execution of the test. The process utilized to ensure a representative test scenario was explained to describe the relationship of input parameters, the data transfer network architecture and system response variables. This included definition, documentation and rationale of the test factors, control factors, and system response variables that were utilized in the test execution. Finally, a description of the experimental test design was provided and the experimental procedures explained. The results of the executed experiment and the developed system model and results are the topic of Chapter IV.

IV. Analysis and Results

Chapter Introduction

This chapter documents the analysis of the executed experimental design for the system responses of the data transfer network architecture. This analysis focuses on the application of statistical methods to determine significant factors and interactions of factors that affect the performance of the data transfer network architecture. The data gathered from the experimental test runs focused on the optimization of system response variables often reported in both academic and commercial publications about data transfer network optimization capabilities.

Focus of Analysis

This effort attempts to derive answers to the investigative questions based on responses from an emulated network environment that is representative of the operational data transfer network architecture that employs the UDT data transfer protocol for large-format sensor data transfers. The UDT data transfer protocol was chosen in the operational network to support the transition from a commercial data transfer capability that was no longer supportable on the operational network. Initial testing in a pristine laboratory environment provided promising results for the data transfer capability utilizing UDT, however, upon employment to the operational network, performance degraded significantly using the default settings of the UDT protocol. The analysis from this effort will provide critical insight into areas of optimization for ongoing development efforts to improve the efficiency of the UDT protocol in the defined operational environment where prior academic publications have not addressed performance of the protocol. As described in Chapter II, the high-bandwidth delay product architectures for which UDT was developed are defined by high network bandwidth and low network latency, whereas, the operational network is the inverse of this assumption as the BDP is primarily affected by the high latency component with average network bandwidth.

Network Goodput Utilization

The data transfer network system response variable *Goodput Utilization* is the focus of the analysis performed. This decision was made to ensure recommendations in system performance enable useful utilization of the operational network resources provided for large-format sensor data transfer architecture. In the process of collecting the *Network Goodput* response variable, the *Target Transfer Data Rate* can adversely affect the comparison of the performance metric due to the varying of the input factor between different target settings. The use of the utilization percentage of the *Target Transfer Data Rate* allows for a normalization of results to allow for a direct comparison between differing performance values.

The use of the *Network Throughput* response variable and the associated *Network Throughput Utilization* response variable highlights an additional issue with the performance of a network architecture. As defined in this effort, high network throughput does not necessarily mean that the data transfer protocol is performing at an efficient rate. Should the data transfer protocol be generating many packet retransmissions due to an overly aggressive attempt to consume bandwidth, the underlying layers of the IP stack may be forced to delay or altogether discard data packets being sent, causing the need for a retransmission from the data transfer protocol.

It is for this reason that a high network throughput utilization rate does not necessarily mean the scarce network resources are being utilized efficiently and thus the response variable of *Network Goodput Utilization* is the true response variable of interest in the following analysis.

Results for Network Goodput Utilization

A model was built using JMP for the full factorial design for the response variable *Network Goodput Utilization* with the representative results for Summary of Fit, Analysis of Variance and Parameter Estimates shown in Table 30, Table 31 and Table 32, respectively. Detailed results for raw data are documented in Table 45 and Table 46 of Appendix D - Raw Experimental Data. The calculated system response variables are found in Table 49 and Table 50 of Appendix E - Calculated System Response Variables.

Table 30 - Summary of Fit for 2^4 Design

RSquare	0.980003
RSquare Adj	0.973754
Root Mean Square Error	5.417278
Mean of Response	45.58453
Observations (or Sum Wgts)	64

Table 31 - Analysis of Variance for 2^4 Design

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	15	69035.120	4602.34	156.8255
Error	48	1408.651	29.35	**Prob > F**
C. Total	63	70443.771		<.0001*

Table 32 - Parameter Estimates for 2^4 Design

Term	Estimate	t Ratio	Prob>|t|	Std Beta
Intercept	65.225232	26.91	<.0001*	0
Network Latency	-0.051966	-11.51	<.0001*	-0.23495
Target Transfer Data Rate	-1.394573	-30.89	<.0001*	-0.63052
Rate Control Interval	30.934028	2.06	0.0453*	0.041958
Algorithm Packet Size	0.0382974	32.35	<.0001*	0.660288
Network Latency*Target Transfer Data Rate	0.0020434	6.79	<.0001*	0.138581
Network Latency*Rate Control Interval	-0.116458	-1.16	0.2514	-0.02369
Network Latency*Algorithm Packet Size	5.922e-5	7.50	<.0001*	0.153153
Target Transfer Data Rate*Rate Control Interval	1.1965278	1.19	0.2388	0.024344
Target Transfer Data Rate*Algorithm Packet Size	-0.000433	-5.48	<.0001*	-0.11192
Rate Control Interval*Algorithm Packet Size	0.0606243	2.30	0.0256*	0.047035
Network Latency*Target Transfer Data Rate*Rate Control Interval	-0.003677	-0.55	0.5850	-0.01122
Network Latency*Target Transfer Data Rate*Algorithm Packet Size	-4.355e-6	-8.28	<.0001*	-0.16895
Network Latency*Rate Control Interval*Algorithm Packet Size	-4.812e-5	-0.27	0.7850	-0.0056
Target Transfer Data Rate*Rate Control Interval*Algorithm Packet Size	0.0028972	1.65	0.1051	0.033716
Network Latency*Target Transfer Data Rate*Rate Control Interval*Algorithm Packet Size	-0.000019	-1.63	0.1094	-0.03329

These results document all main and interaction effects from the 2^4 design and demonstrate the fit of the linear model with an adjusted R-squared value of 0.9738. Based on the parameter estimates, the input factors that are significant at the p = 0.05 level are: *Network Latency*, *Target Transfer Data Rate*, *Rate Control Interval*, *Algorithm Packet Size*, the two-way interaction effects of *Network Latency * Target Transfer Data Rate*, *Network Latency * Algorithm Packet Size*, *Target Transfer Data Rate * Algorithm Packet Size*, *Rate Control Interval * Algorithm Packet Size*, and the three-way interaction effect of *Network Latency * Target Transfer Data Rate * Algorithm Packet Size*. Prior to further analysis, the model was reduced to remove the factors that are not significant at the 0.05-level to allow for improved interpretability of the significant main and interaction effects on the system response variable of *Goodput Utilization*.

The significant factors to be included in the model for *Goodput Utilization*, in sorted order of effect as interpreted by the magnitude of the standardized regression coefficients, are as follows:

Algorithm Packet Size
Target Transfer Data Rate
Network Latency
*Network Latency * Target Transfer Data Rate * Algorithm Packet Size*
*Network Latency * Algorithm Packet Size*
*Network Latency * Target Transfer Data Rate*
*Target Transfer Data Rate * Algorithm Packet Size*
*Rate Control Interval * Algorithm Packet Size*
Rate Control Interval

These factors were selected due to their significance at the p = 0.05 level for their effect on the response variable from the initial 2^4 design results. The sorted results of the model based on the parameter estimate are shown in Table 33.

Table 33 - Sorted Parameter Estimates for Regressed Model

Term	Estimate	t Ratio	Prob>|t|	Std Beta
Algorithm Packet Size	0.0382974	31.62	<.0001*	0.660288
Target Transfer Data Rate	-1.394573	-30.19	<.0001*	-0.63052
Network Latency	-0.051966	-11.25	<.0001*	-0.23495
Network Latency*Target Transfer Data Rate*Algorithm Packet Size	-4.355e-6	-8.09	<.0001*	-0.16895
Network Latency*Algorithm Packet Size	5.922e-5	8.076e-6	<.0001*	0.153153
Network Latency*Target Transfer Data Rate	0.0020434	6.64	<.0001*	0.138581
Target Transfer Data Rate*Algorithm Packet Size	-0.000433	-5.36	<.0001*	-0.11192
Rate Control Interval*Algorithm Packet Size	0.0606243	2.25	0.0284*	0.047035
Rate Control Interval	30.934028	2.01	0.0495*	0.041958

The effect of removing the non-significant terms from the model is seen in the slight decrease of the adjusted R-squared value from 0.9734 to 0.9725 as shown in Table 30 and Table 34, respectively. This effect is also seen in the Analysis of Variance, shown in Table 35, where the Sum of Squares of the non-significant terms are added to the Error component of the model.

Table 34 - Summary of Fit for Regressed Model

RSquare	0.976447
RSquare Adj	0.972521
Root Mean Square Error	5.543078
Mean of Response	45.58453
Observations (or Sum Wgts)	64

Table 35 - Analysis of Variance for Regressed Model

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	9	68784.583	7642.73	248.7406
Error	54	1659.188	30.73	**Prob > F**
C. Total	63	70443.771		<.0001*

Based on the significant factors, the resulting model for *Goodput Utilization* is shown in Equation 15.

$$\begin{aligned} \mathit{Goodput\ Utilization} &= 65.2252 - 0.051966 * \mathit{Network\ Latency} - 1.39457 \\ &* \mathit{Target\ Transfer\ Data\ Rate} + 30.9340 * \mathit{Rate\ Control\ Interval} \\ &+ 0.038297 * \mathit{Algorithm\ Packet\ Size} + 0.002043 \\ &* (\mathit{Network\ Latency} * \mathit{Target\ Transfer\ Data\ Rate}) + 0.000059 \\ &* (\mathit{Network\ Latency} * \mathit{Algorithm\ Packet\ Size}) - 0.000433 \\ &* (\mathit{Target\ Transfer\ Data\ Rate} * \mathit{Algorithm\ Packet\ Size}) \\ &+ 0.060624 * (\mathit{Rate\ Control\ Interval} * \mathit{Algorithm\ Packet\ Size}) \\ &- 0.000004 \\ &* (\mathit{Network\ Latency} * \mathit{Target\ Transfer\ Data\ Rate} \\ &* \mathit{Algorithm\ Packet\ Size}) \end{aligned}$$

Equation 15 - Model of System Performance

Main Effects

Several observations are apparent from the model and the model estimates as sorted by the standardized coefficients. The *Target Transfer Data Rate* and the *Algorithm Packet Size* have a large impact on system performance. The standardized coefficients of these factors indicate that *Goodput Utilization* and *Algorithm Packet Size* are proportionally related, while *Target Transfer Data Rate* and *Goodput Utilization* are inversely related. Based on this logic, the choice of *Algorithm Packet Size* should be to choose the high setting, or 1400 byte packet size, while the *Target Transfer Data Rate* should be set at its low setting of 10 Mbps in order to maximize the *Goodput Utilization* of the system.

The selection of *Algorithm Packet Size* is intuitive if one considers the associated network overhead that is associated with the selection of small packet sizes. The choice of smaller packet sizes decreases the percentage of packet overhead to data within a data packet. Based upon the UDT protocol this ratio is 0.9% for the 1400-byte data packet (12 bytes of UDT packet overhead for each 1388 bytes of data) vice 4.9% for the 256-byte data packet (12 bytes of UDT packet overhead for each 244 bytes of data), with a delta of 4%. In comparison, the *Algorithm Packet Size* factor accounts for an approximately 20% change in utilization of network resources for goodput.

The selection of the *Target Transfer Data Rate* to improve *Goodput Utilization* counters standard thought that increased data transfer rates lead to improved transfer of data via a data transfer network architecture. In order to improve the utilization rate of the actual data being transferred via the network, the low setting of 10 Mbps should be chosen. This is likely due to the fact that attempting to drive the network at its maximum transfer rate actually diminishes the ability of the underlying network architecture to perform, thus packet loss or retransmissions within the network occur at a greater rate, thus reducing goodput.

The selection of *Network Latency* follows standard intuition of less latency in a network architecture providing better performance of data transfer algorithms and the low setting of 200 ms is desired in order to maximize *Goodput Utilization*.

The selection of the *Rate Control Interval* is proportional to the system response thus the high setting of 0.1s produces a higher rate of *Goodput Utilization*.

These model-based observations are apparent in the prediction profile graphic in Figure 18. The negative slope of the profile plot for *Network Latency* and *Target Transfer Data Rate* correlate to the inverse relationship to the modeled system response variable. Similarly, the positive slope of the profile plot for *Algorithm Packet Size* and *Rate Control Interval* profiles correlate to the proportional relationship to the modeled system response variable. The slope of the profile plot indicates the magnitude of the effect for the respective input variable on the system response.

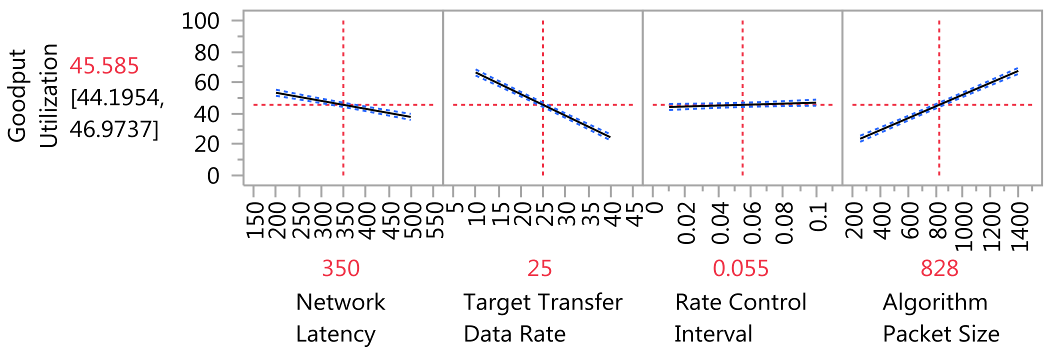


Figure 18 - Prediction Profile for Model of *Goodput Utilization*

The optimized selection of input factors to maximize *Goodput Utilization* within the bounds of the system model is shown in Table 36.

Table 36 - Optimized Factor Settings

Factor	Discrete Setting
Algorithm Packet Size	1400 bytes
Target Transfer Data Rate	10 Mbps
Network Latency	200 ms
Rate Control Interval	0.1 sec

The optimized system profile is shown in Figure 19 that was created using the JMP prediction profiler.

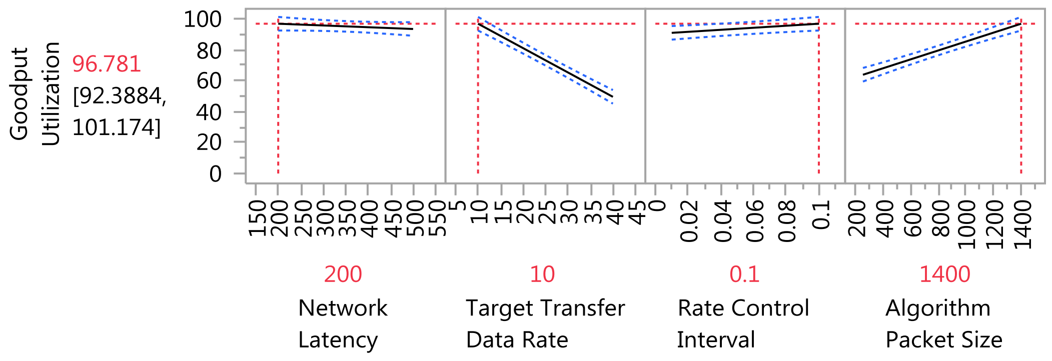


Figure 19 - Optimized System Profile

The association of the interaction effects to their effect on the system response is apparent from the JMP Interaction Profiles shown in Figure 20. Most notable is the change in the effect of *Network Latency* on the system response between the discrete settings of *Algorithm Packet Size*. When the *Algorithm Packet Size* is set to small packets segment sizes, the effect of *Network Latency* is greater, as observed by the increase in the magnitude of the slope of the interaction profile plot. A similar interaction relationship exists between the factors of *Network Latency* and *Target Transfer Data* Rate.

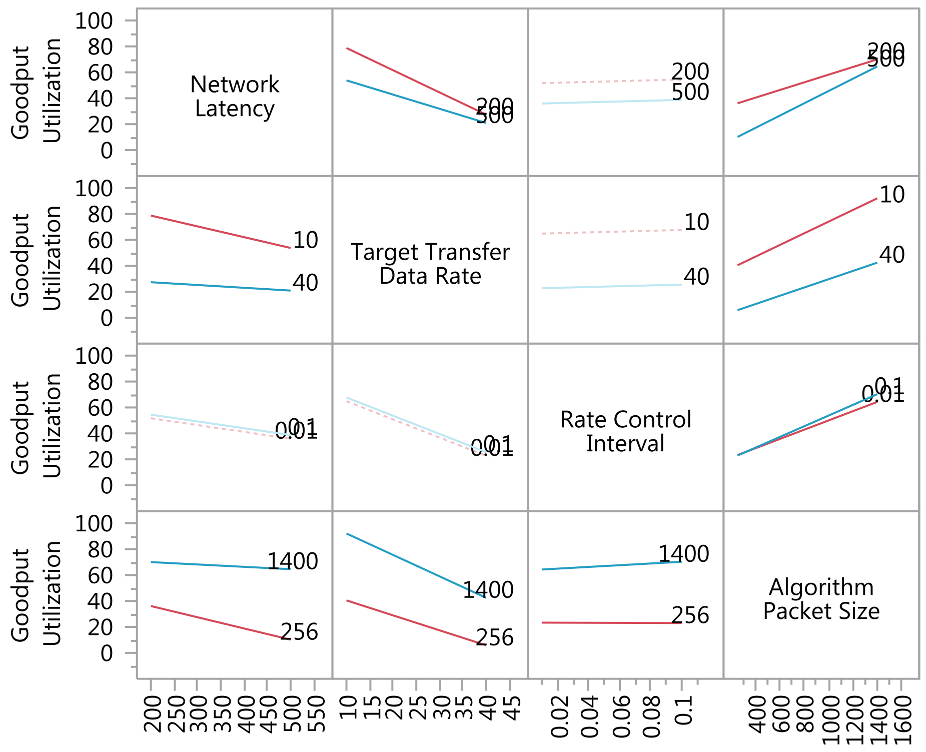


Figure 20 - Model Interaction Profiles

Summary

This chapter documented the selection and analysis of the executed experimental design for the system response of *Goodput Utilization*. The analysis capitalized on the lessons gleaned from the pilot study and screening design described in Chapter III and produced a model of system performance. The model allowed for the recommendation of an optimal configuration of the network architecture parameter of *Network Latency* and of the UDT data transfer protocol parameters of *Algorithm Packet Size, Target Transfer Data Rate and Rate Control Interval*. The information gathered from the analysis of the experimental data highlighted performance observations of the UDT data transfer protocol outside of those referenced in previous research.

V. Conclusions and Recommendations

Chapter Introduction

The focus of this research was to perform a thorough analysis of a specific network architecture and application layer data transfer capability to enable optimized transfer of large format sensor data between remote and local storage architectures. This analysis documented the variables within the architecture and provided a baseline model for performance of the network architecture for current and future data transfer capabilities. This model enables timely analysis of the documented factors that affect the data transfer network architecture and demonstrated the ability to develop a structured test design for this problem set to move beyond the time and manpower inefficiencies of one-factor-at-a-time test strategies. This chapter discusses the impact of the analysis performed in Chapter IV and the resultant answers to the research questions. Finally, conclusions and recommendations for future areas of research related to operational data transfer network architecture development and operations will be proposed.

Applicability to Research Questions

The research documented in this thesis focused on both academic and commercial optimization techniques utilized for improved data transfer through network architectures and capitalized on this foundation to determine the true factors that influence the performance of the multi-tiered computer network utilized to transmit large format sensor data for analysis and production. Through the creation of a representative network architecture, based on modeled performance through network emulation, this research has answered the research questions in the following sections.

Research Question 1.

What are the optimal application layer protocol configurations for a specific large-format sensor data transfer architecture?

This effort focused on the use of UDT in a specific large-format sensor data transfer architecture but the overall process is applicable to any implementation of a data transfer protocol employed. Based on the data collected via the methodology presented in Chapter III and the analysis conducted in Chapter IV, the optimal configuration of the protocol was determined for the design space. The screening design of the pilot study highlighted that the *Transfer File Size* did not have a significant impact of the *Goodput Utilization* of the data transfer network architecture under test. The model created for the response variable of *Goodput Utilization* enabled the assessment of the application-layer protocol input factors of *Target Transfer Data Rate, Algorithm Packet Size* and *Rate Control Interval*. Based on the design space, the optimal configuration of these protocol parameters is shown in Table 37, where the design space is limited to values between the discrete settings.

Table 37 - Algorithm Configuration for Goodput Utilization within Design Space

Factor	**Discrete Settings**		**Optimal Configuration**
	High (+)	**Low (-)**	
Target Transfer Data Rate	40 Mbps	10 Mbps	10 Mbps
Algorithm Packet Size	1400 bytes	256 bytes	1400 bytes
Rate Control Interval	0.1 ms	0.01 ms	0.1 ms

Research Question 2.

What are the factors that most greatly affect the performance of this data transfer architecture?

As derived from the model developed in Chapter IV for *Goodput Utilization*, the factors that most greatly affect the performance of the tested data transfer architecture are *Algorithm Packet Size*, *Target Transfer Data Rate*, *Network Latency* and *Rate Control Interval*. The use of a screening design was able to limit the initial six input factors to the four above thus reducing the experimental design space and allowing for focused research on those factors of greatest importance. Due to the utilization of a structured test design process and the creation of a representative emulative environment in which to test, the answer to this research question was derived from a straight-forward and repeatable process. Through the review of the magnitude of the standardized coefficients and the use of graphical representations such as the prediction profiler, the "order of effect" of the factors allows the researcher to quickly sort and prioritize the results. In fact, the effect of *Algorithm Packet Size* and *Target Transfer Data Rate* is nearly three times the importance of *Network Latency* and fifteen times the effect of the *Rate Control Interval*.

Research Question 3.

What is the expected performance of the data transfer architecture based upon the developed model for the current employment?

The model generated in Chapter IV for the response variable *Goodput Utilization* can be applied to the current operational employment of the data transfer architecture through the use of representative parameters of the operational network to derive an estimate of the expected performance. The screening designs results indicated that expected result for *Goodput Utilization* is consistent across the design space tested for the network architecture parameter of *Packet Loss Rate* for results between 0% and 5%.

Likewise, the algorithm input factor of *Transfer File Size* holds this same assumption for file sizes of 5 MB to 300 MB. Due to this fact, these factors can be set to those that represent the state of the operational network.

The model for *Goodput Utilization* takes into account the network parameter of *Network Latency* across the range of 200 ms to 500 ms with as the bounds of the input factor. As the operational network architecture allows for no controls of quality of service and very limited inputs into quality of performance (i.e. bandwidth and network latency), the developed model can be used to estimate performance due to the fact that the discrete settings for the *Network Latency* factor were chosen to include the observed performance of the specific operational network. Through use of tools, such as the JMP prediction profiler, the specific setting for *Network Latency* can be dialed-in while the remaining factors *Algorithm Packet Size*, *Target Transfer Rate* and *Rate Control Interval* are set to their recommended values, enabling the ability to obtain an estimate of system performance.

Applicability to Related Endeavors

Completion of this research effort addresses these additional questions related to test and evaluation efficiencies and operational architecture limitations:

What efficiencies in deployment of future capabilities for optimized data transfer might be realized through the utilization of a structured test design approach?

As demonstrated in this effort, the thorough analysis of the system architecture documents the ability to discover factors of interest in the development, employment and optimization of existing or future data transfer capabilities. Close integration with the application developer would allow for improved understanding of the capability and allow for coordination between the co-development of the structured test design and the data transfer capability in question. This enables timely feedback to prioritize limited development resources to those areas that provide the largest anticipated effect on the system responses of interest. Through the replacement of one-factor-at-a-time design and test strategies with efficient and timely test design strategies, improved performance of data transfer network architectures can be attained.

Is there any justification available to levy additional requirements on the administratively removed sections of the system architecture that would enable further areas of optimization for data transfer network architectures?

As seen in the analysis of the emulative system there appears to be sufficient justification to support additional requirements on the network architecture to improve *Network Latency* in order to enable more efficient transfer of data through the network. The inverse relationship between the response variable of interest, *Goodput Utilization*, and the input factor of *Network Latency* indicates that to improve system response, one must decrease, therefore improve, latency in the architecture. This is true for smaller data packet sizes, as seen by the interaction with *Algorithm Packet Size*, and for higher network transmission rates, as seen by the interaction with *Target Transfer Rate*.

Recommendations for Future Action

The direct applicability of this research effort to existing operational data transfer network architecture for large-format data sensors enables improved utilization of limited network resources. As described in the previous section, the lessons learned from the established emulative environment should be used to support a vectoring of ongoing development and optimization efforts for the operational capability. Based on intelligent planning, a main factor experiment should be performed on the operational system to begin the validation process of the emulative environment, improve the development effort and ultimately attempt to reduce the requirement on scarce network resources in the operational environment.

The potential system response variables of Sending Cost and Receiving Cost were unable to be accurately collected within the scope of this effort. Future development and ongoing enhancement efforts should consider planning for the necessary data collection points to enable collection of the appropriate raw data to calculate these statistics as they can provide additional insight into the operations of the network architecture through their relationship to what the data sender and receiver are attempting to accomplish during the data transfer. The use of these system responses by commercial applications shows their applicability to the market and make for a clear and consistent metric to compare data transfer technologies in an Analysis of Alternatives process or during system development and fielding.

Recommendations for Future Research

Due to ongoing operations, the network architecture for future systems is already morphing from the architecture design space utilized in this research. Existing network architectures can still benefit from this research, but the future network constraints required the analysis of a larger parameter space for network architecture performance, especially in relation to deployment of capabilities to even more disparate operational locations but with increasing demand for data access and retrieval via data transfer networks.

The use of a Definitive Screening Design allowed for an assumption of a linear response model for the system response of interest. Investigation into expanded parameter areas or into smaller parameter areas at, or near, the boundaries of operational data transfer architectures may lend towards the need to develop non-linear models to more accurately assess system performance.

This effort focused on an average of system response and statistics gathered over a period of time based on data transfer of a file of a size representative of the operational data transfer architecture. Future research may focus on transmission of actual data files that vary in size and delivery order to assess performance of the data transfer mechanism in that environment.

Expanded research could be conducted on a transition from pure data transfer as the primary use of the network architecture to support additional data access paradigms represented by remote system access or focused data streaming of only the data required by the analyst to perform their specific task. These areas shift the network paradigm from a steady-state transfer paradigm towards a more unstructured transfer process that ebbs and flows with analyst utilization.

A comparison between different data transmission mechanisms can benefit from a structured test design approach. This effort focused on the as-employed network data transfer architecture and end-to-end technical solution. Future studies should investigate alternative application layer applications and employ the test design methodology developed in the assessment of existing capabilities or the development of new capabilities. The focus of the UDT implementation in this effort had its basis in pipelining of data transfer. Future efforts could focus on the application implementation of UDT to study its use in parallel and concurrent data transfer methodologies.

Lastly, this study did not focus on the optimization of the client and server hardware architecture for the employed system as it was assumed that these resources were not limited when compared to the data transfer network architecture. As the related research highlighted, much higher speed data transfer network architectures are in use for this very purpose. Further efforts exploring this regime of network architectures may likely require investigation into optimization of the client and server hardware architecture to support efficient utilization of the data transfer algorithm.

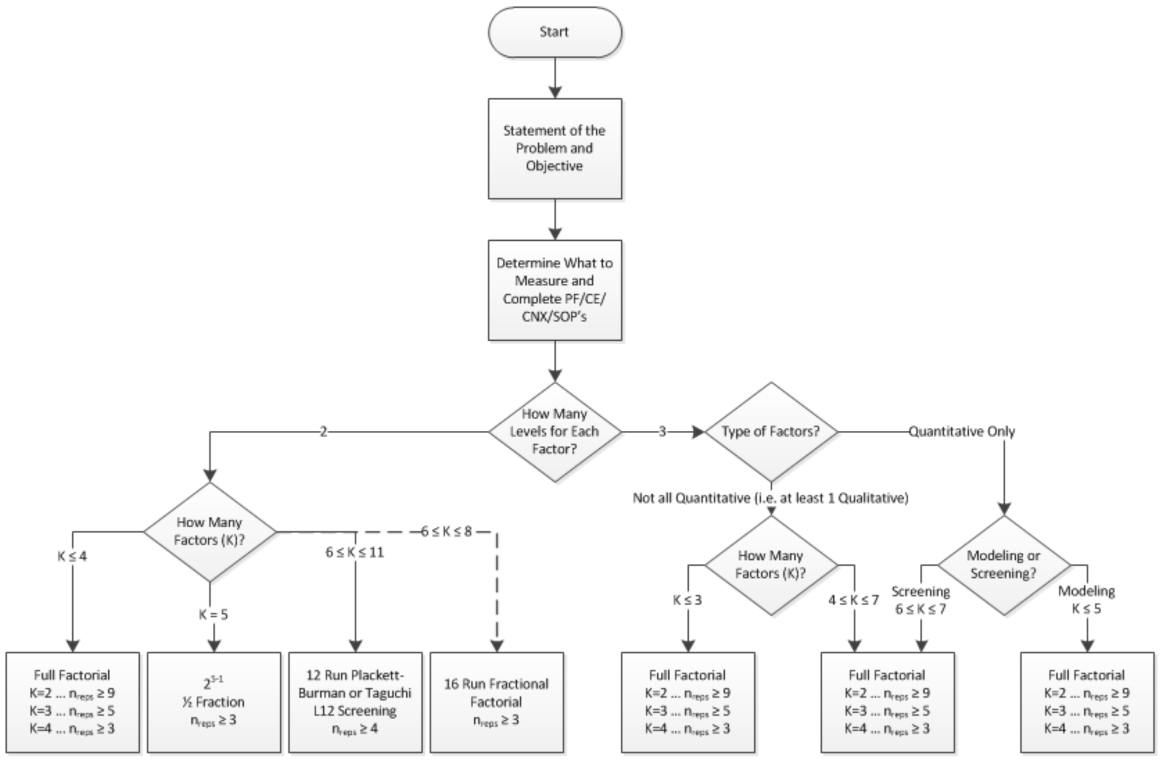
Summary

This research demonstrated the ability to document, model and analyze a data transfer network architecture and the employed data transfer protocol with sufficient rigor to justify recommended modifications to an operational data transfer network architecture. The recommended modifications from this effort will enable the efficient utilization of scarce network resources between remote and local storage architectures that support transmission and receipt of large format sensor data. The analysis documented a baseline model of system performance that will be used to guide ongoing maintenance, sustainment and enhancement efforts for the current data transfer capability and provides insight into the recommended test design process for use in development and deployment of future capabilities. The ability to model system performance through the use of a structured and straight-forward process allows for the inclusion of the test design and analysis process in software design and development, as well as, system deployment and operations improvements.

Appendix

Appendix A – Experimental Design Guidelines

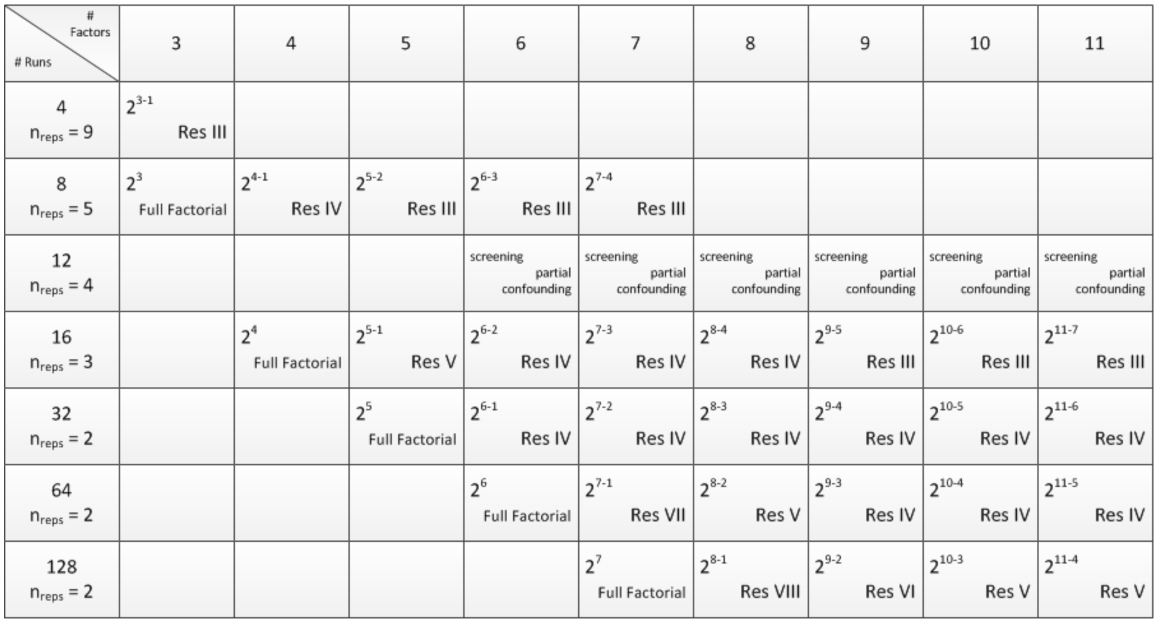
Figure 21 shows the Keep It Simple Statistically (KISS) Rule of Thumb Guidelines for Choosing an Experimental Design as described by Schmidt (2005).


Start
Statement of the Problem and Objective
Determine What to Measure and Complete PF/CE/CNX/SOP's
How Many Levels for Each Factor?
2
3
Type of Factors?
Quantitative Only
Not all Quantitative (i.e. at least 1 Qualitative)
How Many Factors (K)?
K ≤ 4
K = 5
6 ≤ K ≤ 11
6 ≤ K ≤ 8
How Many Factors (K)?
K ≤ 3
4 ≤ K ≤ 7
Modeling or Screening?
Screening 6 ≤ K ≤ 7
Modeling K ≤ 5
Full Factorial K=2 ... n_reps ≥ 9 K=3 ... n_reps ≥ 5 K=4 ... n_reps ≥ 3
2^(5-1) ½ Fraction n_reps ≥ 3
12 Run Plackett-Burman or Taguchi L12 Screening n_reps ≥ 4
16 Run Fractional Factorial n_reps ≥ 3
Full Factorial K=2 ... n_reps ≥ 9 K=3 ... n_reps ≥ 5 K=4 ... n_reps ≥ 3
Full Factorial K=2 ... n_reps ≥ 9 K=3 ... n_reps ≥ 5 K=4 ... n_reps ≥ 3
Full Factorial K=2 ... n_reps ≥ 9 K=3 ... n_reps ≥ 5 K=4 ... n_reps ≥ 3

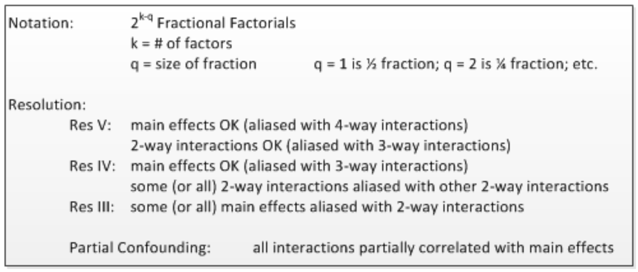
NOTE 1:	Sample size (n_{reps}) is for 95% confidence in estimate of predicted standard deviation and 99.99% confidence in estimate of predicted response
NOTE 2:	(n_{reps}/2) will provide 75% confidence in estimate of predicted standard deviation and 95% confidence in estimate of predicted response
NOTE 3:	The 12 run Plackett-Burman or L12 is very sensitive to large numbers of interactions. If this is the case, you would be better off using the 16 Run Fractional Factorial or a smaller number of variables in 2 or more Full Factorial experiments.

Figure 21 - Guidelines for Choosing an Experimental Design

Figure 22 shows a summary of 2-level designs as described by Schmidt (2005) that expands upon the 2-level designs in Figure 21.



# Factors / # Runs	3	4	5	6	7	8	9	10	11
4 $n_{reps} = 9$	2^{3-1} Res III								
8 $n_{reps} = 5$	2^{3} Full Factorial	2^{4-1} Res IV	2^{5-2} Res III	2^{6-3} Res III	2^{7-4} Res III				
12 $n_{reps} = 4$				screening partial confounding	screening partial confounding	screening partial confounding	screening partial confounding	screening partial confounding	screening partial confounding
16 $n_{reps} = 3$		2^{4} Full Factorial	2^{5-1} Res V	2^{6-2} Res IV	2^{7-3} Res IV	2^{8-4} Res IV	2^{9-5} Res III	2^{10-6} Res III	2^{11-7} Res III
32 $n_{reps} = 2$			2^{5} Full Factorial	2^{6-1} Res IV	2^{7-2} Res IV	2^{8-3} Res IV	2^{9-4} Res IV	2^{10-5} Res IV	2^{11-6} Res IV
64 $n_{reps} = 2$				2^{6} Full Factorial	2^{7-1} Res VII	2^{8-2} Res V	2^{9-3} Res IV	2^{10-4} Res IV	2^{11-5} Res IV
128 $n_{reps} = 2$					2^{7} Full Factorial	2^{8-1} Res VIII	2^{9-2} Res VI	2^{10-3} Res V	2^{11-4} Res V



Notation: 2^{k-q} Fractional Factorials
k = # of factors
q = size of fraction q = 1 is ½ fraction; q = 2 is ¼ fraction; etc.

Resolution:
Res V: main effects OK (aliased with 4-way interactions)
2-way interactions OK (aliased with 3-way interactions)
Res IV: main effects OK (aliased with 3-way interactions)
some (or all) 2-way interactions aliased with other 2-way interactions
Res III: some (or all) main effects aliased with 2-way interactions

Partial Confounding: all interactions partially correlated with main effects

Figure 22 - 2-Level Design Summary

Appendix B – Collection Parameters for Response Variable Calculation

The UDT TRACEINFO structure stores the performance trace information for the protocol. The member attributes can be read directly by user applications. The member attributes available at listed in Table 38 through Table 40 as depicted in Gu (2011). Table 38 depicts aggregate values since the UDT socket was created.

Table 38 - Aggregate UDT Statistics

Members	Comments
int64_t msTimeStamp	time elapsed since the UDT socket is created, in milliseconds
int64_t pktSentTotal	total number of sent packets, including retransmissions
int64_t pktRecvTotal	total number of received packets
int pktSndLossTotal	total number of lost packets, measured in the sending side
int pktRcvLossTotal	total number of lost packets, measured in the receiving side
int pktRetransTotal	total number of retransmitted packets, measured in the sending side
int pktSentACKTotal	total number of sent ACK packets
int pktRecvACKTotal	total number of received ACK packets
int pktSentNAKTotal	total number of sent NAK packets
int pktRecvNAKTotal	total number of received NAK packets

The statistic pktRecvTotal and msTimeStamp will be used to calculate the response variable *Network Throughput* via the relationship shown in Equation 16.

$$Network\ Throughput = \frac{pktRecvTotal\ * Algorithm\ Packet\ Size * 8}{msTimeStamp * \left(\frac{1}{1000}\right)}$$

Equation 16 - Measured Response Variable Network Throughput

Where:
Network Throughput = data rate average over the sending period in Mbps
pktRecvTotal = total number of received packets at the UDT Receiver
Algorithm Packet Size = experimental factor controlling UDT Packet Size
msTimeStamp = time in milliseconds since the UDT socket was created

The statistic msTimeStamp will be used to calculate the response variable *Network Goodput* via the relationship shown in Equation 17.

$$Network\ Goodput = \frac{DataWrittenatReceiver * 8}{msTimeStamp * \left(\frac{1}{1000}\right)}$$

Equation 17 - Measured Response Variable Network Goodput

Where:
Network Goodput = goodput rate average over the sending period in Mbps
DataWrittenatReceiver = file size of data transferred in bytes
msTimeStamp = time in milliseconds since the UDT socket was created

Table 39 depicts local values that are representative of the change in attribute parameters since the last time they were recorded.

Table 39 - Delta UDT Statistics

Members	Comments
int64 pktSent	number of sent packets, including retransmissions
int64 pktRecv	number of received packets
int pktSndLoss	number of lost packets, measured in the sending side
int pktRcvLoss	number of lost packets, measured in the receiving side
int pktRetrans	number of retransmitted packets, measured in the sending side
int pktSentACK	number of sent ACK packets
int pktRecvACK	number of received ACK packets
int pktSentNAK	number of sent NAK packets
int pktRecvNAK	number of received NAK packets
double mbpsSendRate	sending rate in Mbps
double mbpsRecvRate	receiving rate in Mbps

Table 40 depicts instantaneous values that are representative of the current state of the attribute parameters within UDT.

Table 40 - Instantaneous UDT Statistics

Members	**Comments**
double usPktSndPeriod	packet sending period, in microseconds
int pktFlowWindow	flow window size, in number of packets
int pktCongestionWindow	congestion window size, in number of packets
int pktFlightSize	number packets on the flight
double msRTT	round trip time, in milliseconds
double mbpsBandwidth	estimated bandwidth, in Mbps
int byteAvailSndBuf	available sending buffer size, in bytes
int byteAvailRcvBuf	available receiving buffer size, in bytes

Appendix C - Test Design

Table 41 - Full Factorial Test Design (1 of 2)

Run	File Size (MB)	Packet Loss Rate (%)	Network Latency (ms)	Target Transfer Data Rate (Mbps)	Rate Control Interval (sec)	Algorithm Packet Size (bytes)
1	200	2.5	200	10	0.1	256
2	200	2.5	500	10	0.01	256
3	200	2.5	500	10	0.01	1400
4	200	2.5	200	10	0.01	256
5	200	2.5	500	10	0.1	1400
6	200	2.5	500	10	0.1	1400
7	200	2.5	500	10	0.01	1400
8	200	2.5	500	40	0.1	256
9	200	2.5	500	10	0.01	256
10	200	2.5	500	10	0.1	256
11	200	2.5	500	10	0.01	256
12	200	2.5	500	40	0.01	256
13	200	2.5	200	10	0.01	1400
14	200	2.5	200	10	0.1	1400
15	200	2.5	200	40	0.01	1400
16	200	2.5	500	40	0.1	256
17	200	2.5	200	40	0.1	1400
18	200	2.5	200	10	0.1	1400
19	200	2.5	200	40	0.01	1400
20	200	2.5	500	10	0.1	256
21	200	2.5	500	40	0.1	1400
22	200	2.5	500	40	0.1	1400
23	200	2.5	200	10	0.1	1400
24	200	2.5	200	10	0.01	256
25	200	2.5	200	40	0.1	256
26	200	2.5	200	40	0.01	256
27	200	2.5	200	10	0.01	256
28	200	2.5	200	40	0.1	1400
29	200	2.5	500	40	0.1	1400
30	200	2.5	200	40	0.01	1400
31	200	2.5	500	10	0.1	256
32	200	2.5	500	10	0.1	1400
33	200	2.5	200	10	0.1	256
34	200	2.5	200	10	0.01	256
35	200	2.5	200	10	0.01	1400

Table 42 – Full Factorial Test Design (2 of 2)

Run	File Size (MB)	Packet Loss Rate (%)	Network Latency (ms)	Target Transfer Data Rate (Mbps)	Rate Control Interval (sec)	Algorithm Packet Size (bytes)
36	200	2.5	500	40	0.01	1400
37	200	2.5	500	40	0.1	256
38	200	2.5	200	40	0.1	256
39	200	2.5	500	10	0.1	1400
40	200	2.5	200	40	0.01	256
41	200	2.5	500	10	0.01	256
42	200	2.5	200	40	0.1	256
43	200	2.5	200	10	0.1	256
44	200	2.5	500	40	0.01	256
45	200	2.5	500	40	0.01	1400
46	200	2.5	200	10	0.01	1400
47	200	2.5	200	10	0.1	1400
48	200	2.5	500	40	0.01	1400
49	200	2.5	200	10	0.01	1400
50	200	2.5	200	40	0.01	1400
51	200	2.5	500	40	0.1	256
52	200	2.5	500	40	0.1	1400
53	200	2.5	200	40	0.01	256
54	200	2.5	500	10	0.01	1400
55	200	2.5	500	40	0.01	256
56	200	2.5	200	40	0.1	256
57	200	2.5	200	40	0.01	256
58	200	2.5	500	10	0.01	1400
59	200	2.5	500	40	0.01	1400
60	200	2.5	500	40	0.01	256
61	200	2.5	200	40	0.1	1400
62	200	2.5	200	10	0.1	256
63	200	2.5	500	10	0.1	256
64	200	2.5	200	40	0.1	1400

Appendix D - Raw Experimental Data

Raw Data Results

Table 43 - Raw Data from Plackett-Burman Screening Design

Run	pktSentTotal (packets)	msTimeStamp (ms)	Datawritten (bytes)
1	137761	61892	157286400
2	763615	187125	314572800
3	277809	63481	47185920
4	276737	65996	314572800
5	195860	86355	10485760
6	2049837	408666	314573800
7	4946587	1294710	314572800
8	5075217	774652	314572800
9	30682	65667	36700160
10	50323	63792	68157440
11	402479	70616	10485760
12	269640	289489	314572800

Table 44 - Raw Data from Definitive Screening Design

Run	pktSentTotal (packets)	msTimeStamp (ms)	Datawritten (bytes)
1	246226	264593	314572800
2	125420	61161	146800640
3	4473157	1658071	314572800
4	487026	114280	159907840
5	347985	95816	159907840
6	977975	195377	159907840
7	288060	73680	10485760
8	40114	67429	47185920
9	1004814	144780	314572800
10	216283	60471	15728640
11	465136	177796	314572800
12	450293	105395	159907840
13	9542377	1124595	314572800
14	67018	62743	52428800

Table 45 - Raw Data from Full Factorial Design (1 of 2)

Run	pktSentTotal (packets)	msTimeStamp (ms)	Datawritten (bytes)
1	1269044	255283	209715200
2	3182468	881099	209715200
3	166597	181690	209715200
4	1349699	268841	209715200
5	165986	182443	209715200
6	166754	183231	209715200
7	187414	210017	209715200
8	4546825	1183541	209715200
9	3073959	865148	209715200
10	3049107	990801	209715200
11	3243470	962956	209715200
12	6194796	997939	209715200
13	166230	179988	209715200
14	164212	177937	209715200
15	332915	155890	419430400
16	4528584	1223832	209715200
17	332304	147744	419430400
18	164237	177588	209715200
19	166551	180101	209715200
20	3275905	1101765	209715200
21	440281	204475	419430400
22	439789	206086	419430400
23	164390	177739	209715200
24	1343320	266799	209715200
25	3954525	490543	209715200
26	4274317	422630	209715200
27	1352640	269612	209715200
28	331943	156209	419430400
29	394538	200554	419430400
30	166416	179849	209715200
31	2930015	941239	209715200
32	166147	182056	209715200
33	1284245	256255	209715200
34	1337872	268756	209715200
35	166072	179239	209715200

Table 46 - Raw Data from Full Factorial Design (2 of 2)

Run	pktSentTotal (packets)	msTimeStamp (ms)	Datawritten (bytes)
36	166924	182650	209715200
37	4518576	1215689	209715200
38	3819454	472920	209715200
39	165820	182129	209715200
40	4317786	465061	209715200
41	3132310	833329	209715200
42	3992057	502599	209715200
43	1280806	255943	209715200
44	6432265	1000890	209715200
45	423732	205609	419430400
46	166445	179946	209715200
47	164831	178009	209715200
48	415911	196544	419430400
49	166491	180034	209715200
50	333070	148230	419430400
51	4507586	1237803	209715200
52	410129	196427	419430400
53	4405180	419384	209715200
54	166732	182885	209715200
55	6643383	1020292	209715200
56	3925041	501374	209715200
57	4263371	415733	209715200
58	166632	183218	209715200
59	459496	207680	419430400
60	6743017	1026318	209715200
61	332842	161551	419430400
62	1286313	258061	209715200
63	3116868	1017766	209715200
64	332300	157623	419430400

Appendix E - Calculated System Response Variables

Calculated Results

Table 47 - Plackett-Burman Screening Calculated System Response

Run	Network Throughput (Mbps)	Network Goodput (Mbps)	Network Throughput Utilization (%)	Network Goodput Utilization (%)
1	24.93	20.33	49.86	40.66
2	45.70	13.45	91.41	26.90
3	8.96	5.95	89.63	59.46
4	46.96	38.13	93.93	76.26
5	4.65	0.97	9.29	1.94
6	10.27	6.16	102.73	61.58
7	7.82	1.94	78.25	19.44
8	13.42	3.25	26.84	6.50
9	5.23	4.47	52.33	44.71
10	8.84	8.55	88.35	85.47
11	11.67	1.19	23.35	2.38
12	10.43	8.69	104.32	86.93

Table 48 - Definitive Screening Design Calculated System Response

Run	Network Throughput (Mbps)	Network Goodput (Mbps)	Network Throughput Utilization (%)	Network Goodput Utilization (%)
1	10.42	9.51	104.23	95.11
2	22.97	19.20	45.93	38.40
3	5.53	1.52	55.25	15.18
4	28.23	11.19	94.10	37.31
5	40.68	13.35	81.35	26.70
6	10.25	6.55	102.51	65.48
7	8.01	1.14	16.01	2.28
8	6.66	5.60	66.63	55.98
9	45.97	17.38	91.94	34.76
10	7.32	2.08	24.42	6.94
11	29.30	14.15	97.67	47.18
12	28.30	12.14	94.34	40.46
13	17.38	2.24	34.76	4.48
14	7.08	6.68	70.75	66.85

Table 49 - Full Factorial Calculated System Response (1 of 2)

Run	Network Throughput (Mbps)	Network Goodput (Mbps)	Network Throughput Utilization (%)	Network Goodput Utilization (%)
1	10.18	6.57	101.81	65.72
2	7.40	1.90	73.97	19.04
3	10.27	9.23	102.70	92.34
4	10.28	6.24	102.82	62.41
5	10.19	9.20	101.90	91.96
6	10.19	9.16	101.93	91.56
7	9.99	7.99	99.95	79.89
8	7.87	1.42	19.67	3.54
9	7.28	1.94	72.77	19.39
10	6.30	1.69	63.03	16.93
11	6.90	1.74	68.98	17.42
12	12.71	1.68	31.78	4.20
13	10.34	9.32	103.44	93.21
14	10.34	9.43	103.36	94.29
15	23.92	21.52	59.80	53.81
16	7.58	1.37	18.95	3.43
17	25.19	22.71	62.98	56.78
18	10.36	9.45	103.58	94.47
19	10.36	9.32	25.89	23.29
20	6.09	1.52	60.89	15.23
21	24.12	16.41	60.29	41.03
22	23.90	16.28	59.75	40.70
23	10.36	9.44	103.59	94.39
24	10.31	6.29	103.12	62.88
25	16.51	3.42	41.28	8.55
26	20.71	3.97	51.78	9.92
27	10.27	6.22	102.75	62.23
28	23.80	21.48	59.50	53.70
29	22.03	16.73	55.08	41.83
30	10.36	9.33	25.91	23.32
31	6.38	1.78	63.75	17.82
32	10.22	9.22	102.21	92.15
33	10.26	6.55	102.64	65.47
34	10.19	6.24	101.95	62.43
35	10.38	9.36	103.77	93.60

Table 50 - Full Factorial Calculated System Response (2 of 2)

Run	Network Throughput (Mbps)	Network Goodput (Mbps)	Network Throughput Utilization (%)	Network Goodput Utilization (%)
36	10.24	9.19	25.59	22.96
37	7.61	1.38	19.03	3.45
38	16.54	3.55	41.35	8.87
39	10.20	9.21	101.97	92.12
40	19.01	3.61	47.54	9.02
41	7.70	2.01	76.98	20.13
42	16.27	3.34	40.67	8.35
43	10.25	6.56	102.49	65.55
44	13.16	1.68	32.90	4.19
45	23.08	16.32	57.70	40.80
46	10.36	9.32	103.60	93.23
47	10.37	9.42	103.71	94.25
48	23.70	17.07	59.25	42.68
49	10.36	9.32	103.57	93.19
50	25.17	22.64	62.92	56.59
51	7.46	1.36	18.65	3.39
52	23.38	17.08	58.46	42.71
53	21.51	4.00	53.78	10.00
54	10.21	9.17	102.11	91.74
55	13.34	1.64	33.34	4.11
56	16.03	3.35	40.08	8.37
57	21.00	4.04	52.51	10.09
58	10.19	9.16	101.86	91.57
59	24.78	16.16	61.95	40.39
60	13.46	1.63	33.64	4.09
61	23.08	20.77	57.69	51.93
62	10.21	6.50	102.08	65.01
63	6.27	1.65	62.72	16.48
64	23.61	21.29	59.03	53.22

Bibliography

Allman, M., Paxson, V., & Blanton, E. (2009, September). TCP Congestion Control. *RFC 5681*.

An, D., Park, J., Wang, G., & Cho, G. (2012, February 1-3). An Adaptive UDT Congestion Control Method with Reflecting of the Network Status. *International Conference on Information Networking*, 492-496.

Aspera an IBM(R) Company. (2015, 11 29). *Aspera an IBM(R) Company*. Retrieved from Aspera Web site: http://asperasoft.com/company/why-aspera/

Aspera an IBM(R) Company. (2015, November 24). *Aspera FASP(tm) High Speed Transport: A Critical Technology Comparison - White Paper*. Retrieved November 24, 2015, from Aspera fasptm High-speed Transport: http://asperasoft.com/resources/white-papers/aspera-fasptm-high-speed-transport

Barkmo, L. S., & Peterson, L. L. (1995, October). TCP Vegas: End to End Congestion Avoidance on a Global Internet. *IEEE Journal on Selected Ares in Communications*, *13*(8), 1465-1480.

Clark, D. D., & Fang, W. (1998, August). Explicit Allocation of Best Effort Packet Delivery Service. *IEEE/ACM Transactions on Networking, 6*, 362-373.

Colombi, J., Miller, M. E., Schneider, M., McGrogan, J., Long, D. S., & Plaga, J. (2012). Predictive mental workload modeling: implications for system design. *Journal of Systems Engineering, 15*(4), 448-460.

Dordal, P. L. (2014). *An Introduction to Computer Networks.* Shabbona, Illinois: Loyola University Chicago.

Dubie, D. (2004, June 1). WAN Optimization on the Rise. *Network World*.

Eckart, B., He, X., & Wu, Q. (2008, April 14-18). Performance Adaptive UDP for High-Speed Bulk Data Transfer over Dedicated Links. *IEEE International Symposium on Parallel and Distributed Processing*, 1-10.

Eckart, B., He, X., Wu, Q., & Xie, C. (2010, January). A Dynamic Performance-Based Flow Control Method for High-Speed Data Transfer. *IEEE Transactions on Parallel and Distributed Systems, 21*(1), 114-125.

Greico, L. A., & Mascolo, S. (2005). Mathematical Analysis of Westwood+ TCP Congestion Control. *IEE Proceedings - Control Theory Applications. 152*, pp. 35-42. IEE.

Grossman, R. L., Gu, Y., Hong, X., Antony, A., Blom, J., Dijkstra, F., & de Laat, C. (2005). Teraflows over Gigabit WANs with UDT. *Future Generation Computer Systems, 21*, 501-503.

Gu, Y. (2011, February 8). *UDT Manual*. Retrieved from UDT: UDP-based Data Transfer Library - version 4: http://udt.sourceforge.net/udt4/index.htm

Gu, Y., & Grossman, R. L. (2007). UDT: UDP-based data transfer for high-speed wide area networks. *Computer Networks 51*, 1777-1799.

Gu, Y., Hong, X., & Grossman, R. L. (2004). An Analysis of AIMD Algorithms with Decreasing Increases. *First Workshop on Networks for Grid Applications.* San Jose: Gridnets 2004.

Hacker, T. J., Athey, B. D., & Noble, B. (2001, April 15-19). The End-to-End Performance Effects of Parallel TCP Sockets on a Lossy Wide-Area Network. *Proceedings of the International Parallel and Distributed Processing Symposium, IPDPS 2002*.

Kurose, J. F. (2005). *Computer Networking: A Top-Down Approach Featuring the Internet* (3rd Edition ed.). Boston, MA, USA: Pearson Education, Inc.

Liu, S., Basar, T., & Srikant, R. (2008, June). TCP-Illinois: A Loss- and Delay-Based Congestion Control Algoritm for High-Speed Networks. *Performance Evaluation, 65*(6-7), 417-440.

Lu, X., Wu, Q., Rao, N. S., & Wang, Z. (2010, December 6-10). On Parallel UDP-Based Transport Control over Dedicated Connections. *IEEE Global Telecommunications Conference*, 1-5.

Mascolo, S., Casetti, C., Gerla, M., Sanadidi, M. Y., & Wang, R. (2001). TCP Westwood: Bandwidth Estimation for Enhanced Transport Over Wireless Links. *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking (MobiCom '01)* (pp. 287-297). New York: Association for Computing Machinery.

Mathis, M., Semke, J., Mahdavi, J., & Ott, T. (1997, July). The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm. *ACM SIGCOMM Computer Communication Review, 27*(3), 67-82.

Miess, M. R. (n.d.). Tsunami: A High-Speed Rate-Controlled Protocol for File Transfer.

Montgomery, D. C. (2009). *Design and Analysis of Experiments* (7th Edition ed.). Hoboken, NJ, USA: Wiley & Sons, Inc.

Nambiar, M., Kalita, H. K., Mishra, D., Rane, S., Pardeshi, P., & TATA Consultancy Services, L. (. (2014, February 21). *WANem: The Wide Area Network emulator*. Retrieved from WANem: The Wide Area Network emulator: http://wanem.sourceforge.net/

Natrella, M. (2015, November 29). National Institute of Standards and Technology (NIST)/Semiconductor Manufacturing Technology (SEMATECH) e-Handbook of Statistical Methods. Retrieved from http://www.itl.nist.gov/div898/handbook/

nsnam. (2014, December 19). *ns-2: Main Page*. Retrieved from Sourceforge.net: http://nsnam.sourceforge.net/wiki/index.php/Main_Page

Park, J., An, D., & Cho, G. (2013, January 28-30). An Adpative Channel Number Tuning Mechanism on Parallel Transfer with UDT. *International Conference on Information Networking*, 346-350.

PERC - TATA Consultancy Services, Ltd. (2014). *WANem: The Wide Area Network emulator*. Retrieved from WANem: The Wide Area Network emulator: http://wanem.sourceforge.net/

Postel, J. (1980, August). User Datagram Protocol. *RFC 768*.

Postel, J. (1981, September). Transmission Control Protocol. *RFC 793*.

Rizzo, L. (n.d.). *The dummynet project*. Retrieved from Dummynet: http://info.iet.unipi.it/~luigi/dummynet/

SAS Institute Inc. (2015). JMP 12 Design of Experiments Guide. *JMP 12 Online Documentation*. Cary, NC, USA. Retrieved from http://www.jmp.com/en_us/support/jmp-documentation.html

Schmidt, S. R. (2005). *Understanding Industrial Designed Experiments* (4th Edition ed.). Colorado Sprints, CO, USA: Air Academy Press.

Sharma, S., Gillies, D., & Feng, W.-c. (2010). On the Goodput of TCP NewReno in Mobile Networks. *2010 Proceedings of the 19th International Conference on Computer Communications and Networks (ICCCN)*, (pp. 1-8). Zurich.

The LINUX Information Project. (2005, October 16). *Data Link Layer Definition*. Retrieved November 22, 2015, from The LINUX Information Project: http://www.linfo.org/data_link_layer.html

The LINUX Information Project. (2005, September 16). *Network Layer Definition*. Retrieved November 22, 2015, from The LINUX Information Project: http://www.linfo.org/network_layer.html

The LINUX Information Project. (2005, September 29). *Physical Layer Definition*. Retrieved November 22, 2015, from The LINUX Information Project: http://www.linfo.org/physical_layer.html

The Regents of the University of California, through Lawrence Berkeley National Laboratory. (2014). *iperf3: A TCP, UDP, and SCTP network bandwidth measurement tool*. Retrieved from GitHub esnet / iperf: https://github.com/esnet/iperf

Tkaczewski, J. (2010). *Accelerating File Transfers: Increase File Transfer Speeds in Poorly-Performing Networks.* Retrieved from Unlimi-Tech Software, Inc. | FileCatalyst Web site: http://www.filecatalyst.com

Tkaczewski, J. (2012). *Open Source File Transfers: A comparison of recent open source file transfer projects.* Retrieved from Unlimi-Tech Software, Inc. | FileCatalyst Web site: http://www.filecatalyst.com

Totaro, M. W., & Perkins, D. D. (2005). Using Statistical Design of Experiments for Analyzing Mobile Ad Hoc Networks. *Proceedings of the 8th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems* (pp. 159-168). Montreal, Quebec, Canada: ACM.

Utsumi, S., Zabir, S. M., & Shiratori, N. (2008, June 25). TCP-Cherry: A new approach for TCP congestion control over satellite IP networks. *Computer Communications, 31*(10), 2541-2561.

Wei, D. X., Jin, C., Low, S. H., & Hegde, S. (2006, December). FAST TCP: Motivation, Architecture, Algorithms, Performance. *IEEE/ACM Transactions on Networking, 14*(6), 1246-1259.

Xu, K., Tian, Y., & Ansari, N. (2004, May). TCP-Jersey for Wireless IP Communications. *IEEE Journal on Selected Areas in Communications, 22*(4), 747-756.

Yildirim, E., Arslan, E., Kim, J., & Kosar, T. (2015). Application-Level Optimization fo Big Data Transfers Through Pipelining, Parallelism and Concurrency. *IEEE Transactions on Cloud Computing*.

Yu, S.-y., Brownlee, N., & Mahanti, A. (2013, October 21-24). Comparative Performance Analysis of High-speed Transfer Protocols for Big Data. *IEEE Conference on Local Computer Networks*, 292-295.

Zhang, Y., Ansari, N., Wu, M., & Yu, H. (2012). On Wide Area Network Optimization. *IEEE Communications Surveys and Tutorials, 14*(4), 1090-1113.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)* 16-06-2016	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From – To)* March 2010 – June 2016

TITLE AND SUBTITLE Efficient Employment of Large Format Sensor Data Transfer Architectures	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Oltmanns, Jeffrey R., Civilian, Ctr	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way, Building 640 WPAFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT-ENV-MS-16-J-042
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Intentionally left blank)	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRUBTION STATEMENT A. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES
This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT
Due to the increasing quantity of data collected by Air Force intelligence, surveillance and reconnaissance (ISR) assets and the focus on timely access to the data collected by these systems, operational data transfer network architectures have become a critical component of their employment in the intelligence production process. Efficient utilization of the provided long-haul communications component of the ISR system improves the value of the single asset to the warfighter and enables connectivity of additional assets via the data transfer network architecture. To support this evaluation, an emulated network testbed was utilized to develop a representative model of system efficiency. The results of this model indicate that increased aggressiveness for data transfer leads to decreased efficiency in the attempt to utilize available network resources, especially in realm of operations under study that represent a non-traditional bandwidth delay product (BDP) networks where network delay is the dominating factor in the determination of BDP. The analysis documented a baseline model of system performance that will be used to guide ongoing maintenance, sustainment and enhancement efforts for the current data transfer capability and provides insight into the recommended test design process for use in development and deployment of future capabilities.

15. SUBJECT TERMS
Computer Networks, UDP Data Transfer (UDT), Efficient Employment, Emulation, Optimization, Design of Experiments (DoE), Modeling, Regression

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Dr. Brent T. Langhals, AFIT/ENV
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(Include area code)*
U	U	U	UU	132	(937) 255-3636, ext 7402 (brent.langhals@afit.edu)

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18