Communications Middleware for Tactical Environments: Observations, Experiences, and Lessons Learned

Niranjan Suri and Erika Benvegnù, Florida Institute for Human and Machine Cognition
Mauro Tortonesi and Cesare Stefanelli, University of Ferrara
Jesse Kovach, U.S. Army Research Laboratory
James Hanna, U.S. Air Force Research Laboratory

ABSTRACT

Tactical networking environments present significant challenges that must be overcome in order to effectively support net-centric warfare. The wireless and ad hoc nature of these networks implies unreliable connectivity, limited bandwidth, and variable latency. Past and current research has focused on physical and data link layers, routing protocols, transport protocols, and cross-layer aspects. However, significant work is needed at the upper layers to better support application requirements. In our experience, achieving effective communications in tactical environments requires taking into account application requirements and communication patterns, designing a rich interface between the application and communication layers, and realizing a communications middleware specifically adapted to tactical networks. In this article, we report on our observations from several tactical networking experiments and demonstrations and the lessons learned from deployment of the Mockets middleware to support tactical communications. We hope these experiences are useful to others designing and implementing applications and systems for tactical environments.

INTRODUCTION

Network-centric operations [1], also referred to as network-enabled capability and network-based defense, is the cornerstone of modern warfighting. These approaches rely on robust network communications to support timely exchange of information between geographically dispersed entities. However, tactical networks, the basis for network-centric operations, provide one of the most challenging environments for communications (Fig. 1). The inherently mobile nodes must communicate by using wireless ad hoc links in hostile radio frequency (RF) environments, creating unreliable networks that have limited bandwidth and variable latency. Moreover, the dynamic nature of military operations results in widely varying loads being placed on the network by users and applications. Developing effective communication and networking technologies for tactical environments is essential for Network Centric Operations.

Research to date on tactical wireless communications has focused on increasing bandwidth, improving reliability, and enabling adaptation by focusing on areas such as network coding, dynamic spectrum exploitation, robust routing protocols, and cross-layer design. Researchers have also proposed new transport protocols such as the Stream Control Transport Protocol (SCTP) [2], the Transport Protocol of the Space Communications Protocol Standards (SCPS-TP) [3], and TCP enhancements [4]. However, our experience shows that there is much potential for improvement in the realization of communications middleware and application interfaces. Understanding application requirements and communication patterns, and designing protocols to better support them is essential for providing good performance. For example, transport protocols must provide a rich interface to better understand application intent and provide feedback so that applications may adapt to underlying network conditions. Applications in tactical networks have different, sometimes peculiar, requirements; therefore, a one size fits all approach to transport protocols leads to inefficiencies.

This article describes observations and experiences gathered from participating in multiple real-world experiments and demonstrations of net-centric operations in tactical environments in partnership with the U.S. Army Research Laboratory (ARL) and the U.S. Air Force Research Laboratory (AFRL). These include the Horizontal Fusion Quantum Leap experiments in 2003 and 2004 (QL-1 and QL-2), the Communications-Electronics Research, Development, and Engineering Center (CERDEC) C4ISR On the Move (C4ISR OTM) experiments in 2006, 2007, and 2008, and an airborne networking experiment involving a J-STARS platform (ANE). These experiences have driven the design and implementation of Mockets, a communications middleware that supports applications in tactical environments through a rich programming model. Two experiments with Mockets provide a glimpse of the significant performance improvements made possible by the use of a communication framework developed specifically for tactical networks. Mockets demonstrates the importance of tailoring the design and implementation of communications middleware to both the applications and the target environments.

The Tactical Networking Environment: Observations and Lessons Learned

In partnership with ARL and AFRL, we have conducted many experiments in tactical environments, which have resulted in valuable lessons learned. This section summarizes our practical experience, the actual problems encountered with application and radio behavior in tactical network environments, and the associated implications for communications middleware. Note that in this article we do not consider satellite communications.

To better present our observations, we use Blue Force Tracking[1] (BFT) as an example application throughout this article. BFT provides warfighters with location information about friendly military forces. Force XXI Battlefield Command Brigade and Below (FBCB2) is a currently fielded BFT system. A number of other BFT systems are in development.

BFT systems use terrestrial and satellite communication links to exchange GPS position and related situational awareness information. Receiving nodes display the location of friendly forces by using a mapping system. BFT systems can also support other communications, such as text messages and images, as well as information on hostile forces and battlefield obstacles. BFT systems may also be integrated with other battlefield systems such as robotic operator control units (OCUs) and sensor management software. BFT systems operate on real-time information and must satisfy strict latency constraints, as outdated position information can be misleading.

Limited Bandwidth and Intermittent Connectivity

Bandwidth in tactical networks is severely limited, and the throughput achieved in real-world scenarios is often far less than the system specifications would suggest. For example, during the QL-2 experiment, we used a custom 802.11g-based wireless ad hoc IP radio node. 802.11g-based radios provide a theoretical data rate of 54 Mb/s, which decreases to 15–30 Mb/s in a typical office environment. During the experiment we observed an average usable throughput of 1–1.5 Mb/s, which is an order of magnitude lower than expected. The poor performance can be attributed to the high churn rate, RF interference, and zones with poor connectivity.

Figure 1. *Conceptual view of a tactical networking environment.*

Most current and planned military radio systems have even lower bandwidth than 802.11g radio platforms due to additional range, interference, mobility, and security requirements. Experimentation with a prototype of the SLICE radio, an IP radio that implements Joint Tactical Radio Standards (JTRS) Soldier Radio Waveform (SRW), showed a maximum throughput of well under 1 Mb/s shared across all nodes. This radio also has a very high per-packet overhead that significantly reduces the effective bandwidth when an application transmits small packets.

Moreover, application bandwidth requirements are often underestimated. Due in part to poor architectural decisions, the BFT application used for QL-2 consumed approximately 0.5 Mb/s. Therefore, very little bandwidth was available for other tasks such as obtaining sensor data (including video feeds), interpersonal communication, and robot tele-operation (requiring video feeds).

Our experience has also shown that tactical networks are subject to intermittent connectivity problems. In particular, urban environments have many dead spots, which cause nodes to lose connectivity. In a tactical network, where traffic is relayed through multiple hops, the loss of connectivity to one node may affect connectivity to other nodes. Tele-operated robots, increasingly common on the battlefield, present additional complications. A robot that is driven into a dead spot can become stuck due to its inability to receive commands and require manual retrieval, which is not always possible or practical.

Frequent and Abrupt Variations in Channel Conditions and Network Topology

Tactical network environments present turbulent and chaotic network conditions. This stems from the inherent characteristics of radio communication systems including fading, interfer-

[1] In the MIL-STD-2525 symbology standard, friendly forces are designated with blue symbols, enemy forces are designated with red, and unknown forces with yellow.

Applications in tactical networks are extremely heterogeneous, ranging from sensor data dissemination and BFT to robotic teleoperation and Voice over IP (VoIP). Furthermore, data types vary from continuous streams (such as video) to discrete messages (such as individual sensor reports).

ence, and channel contention, resulting in highly variable bandwidth with time and spatial dependencies [5].

Mission execution, responses to enemy activity, and changing tactical objectives cause massive unit movements that are difficult for the network to predict in advance. Network disruption and reconfiguration resulting from this node mobility and churn cause major fluctuations in end-to-end channel conditions. Additional complications arise from velocity differences between air and ground units [6]. As a result, applications that assume continuous connectivity in a *steady state* mode suffer from performance problems and fail in these conditions.

VARIED TYPES OF DATA

Applications in tactical networks are extremely heterogeneous, ranging from sensor data dissemination and BFT to robotic tele-operation and voice over IP (VoIP). Furthermore, data types vary from continuous streams (e.g., video) to discrete messages (e.g., individual sensor reports). Each of these activities potentially demands different types of service from the communications network. We refer to a set of related messages with homogeneous service and message delivery semantics as a data flow. Data flows may require sequential message delivery and/or different reliability levels.

Some data flows contain independent messages and do not have sequencing constraints, while others require message sequencing, typically on a per flow basis. In our experience only a very small number of applications require sequential delivery across all data flows. In addition, some data flows carry critical information that require reliable message delivery, while others carry non-time-sensitive or disposable information that can be sent in a best effort or partially reliable manner. Reliability and sequencing are orthogonal characteristics.

For example, BFT systems deal with separate data flows. Information about units does not have sequencing delivery constraints, but needs partial reliability. Position updates do not have reliability constraints, but require intraflow sequencing as well as sequencing with the last full unit information message. Other examples include sensor reports, which require reliability, and tele-operation and VoIP, which require low latency.

RESOURCE PRIORITIZATION

As a result of reliability, sequencing, and latency requirements, different data flows in a tactical network have different priorities. For example, low-latency flows such as a video feed from a robot have higher priority than sensor reports that will not be viewed in real time. In addition, not all video streams from the robot have equal priority. The video stream for the driver has higher priority than a video stream for an observer, as the driver is vastly more sensitive to latency than the observer. As tactical objectives and environmental conditions change over time, the priority of different data flows will change as well. This was particularly important when we tele-operated a robot using a SLICE radio at the 2007 C4ISR OTM experiment. The radio's constrained bandwidth required us to exploit a feature of the radio where a single high-priority data flow, in this case the driver's video, is sent using a special high-bandwidth mode at the physical layer and takes priority over all other traffic on the network.

DATA QUEUING

When an application exceeds the available network bandwidth, data may accumulate in a transmission queue in the application, operating system, or radio. When these queues become full, new data will be dropped or the application will block until the queue starts to empty.

Data queuing causes temporary spikes and long-term drifts in end-to-end latency along with transmission of unnecessary data. For example, if a node generating BFT information at a rate of 1 Hz loses network connectivity for 30 s, the transmission queue will accumulate 30 messages. When connectivity is restored, all 30 messages will be transmitted in the order they were originally generated. Since the only message of interest is the most recent position update, the preceding 29 messages waste bandwidth and increase latency.

It is extremely difficult for applications to mitigate the impact of data queuing on their overall performance as they have no control over data already in the transmission queue. While careful application design and protocol selection can minimize or eliminate the impact of queuing at the application and operating system levels, this does not eliminate the potential for queuing at the network level.

During the QL-1 experiment, our first attempt used TCP to send all data, including video streams. When the network deteriorated, video latency increased without bound due to the blocking behavior of TCP. The latency of the resulting video stream rendered it useless. During 2007 C4ISR OTM, we encountered a similar problem with the SLICE radio, which contains a very large internal transmit buffer on the order of 1000 packets. Although we had redesigned our applications to minimize buffering and eliminate the use of TCP, the message queuing at the SLICE radio caused problems. The amount of data being transmitted had to be carefully controlled to avoid intolerable latencies.

LEGACY APPLICATIONS AND SYSTEMS

Many legacy applications, protocols, and systems are currently used in tactical environments. These are usually derived from commercial off-the-shelf (COTS) applications developed for the Internet. They use existing protocols such as TCP and UDP, and make assumptions about network services, bandwidth, and availability based on characteristics of the public Internet. These assumptions do not hold for tactical wireless networks, and lead to performance and reliability problems.

The use of TCP is a particularly important problem. TCP was designed for fully-connected wired and wireless infrastructure networks and, as evidenced by its worldwide adoption and success, functions extremely well on these types of networks. However, TCP's congestion control and retransmission behavior was not designed for intermittently connected networks and is often inefficient or too aggressive in these environments [7]. Finally, TCP's (intentionally) simplistic interface does not allow an application to distinguish between different messages and provide additional contextual information (e.g., priority) regarding the messages being sent by the application. While TCP permits transmission of out-of-band high-priority data, this feature provides only two priority levels (normal and high), presents an interface that is not consistent or compatible across platforms, and does not fit in the stream-oriented communication channel abstraction. As a result, this feature of TCP is rarely, if ever, utilized.

Common industry standards such as Web services, HTTP, and some peer-to-peer protocols such as Sun's JXTA make use of TCP. These standards have been adopted by military systems running on higher-echelon (organizations at the level of a brigade and above) command and control networks [8]. TCP functions adequately in such environments, as they are essentially infrastructure networks. However, when these applications are pushed down to the tactical level, the limitations of TCP-based COTS systems become apparent.

Requirements for Communications Middleware in Tactical Networks

Earlier we discussed the challenges posed by tactical wireless environments, which need to be mitigated through explicit support at the middleware level. This section presents a set of requirements for communications middleware to address these challenges. As the resources available are limited, the primary requirement is to enable the application and communications middleware to jointly perform trade-offs between competing demands to improve overall performance.

Application Awareness of Communication State

Tactical networks are characterized by limited bandwidth and intermittent connectivity. This suggests a fundamental requirement for application design: a continuous adaptation process to match both changing tactical objectives and network conditions, instead of assuming steady-state operations. Adaptive application design involves an in-depth reconsideration of the communication stack in order to support preemption, dynamic reallocation, efficient use, and status monitoring of network resources.

Continuous adaptation requires the middleware to support a rich programming model that provides accurate and timely feedback on network status and changes in resource availability to enable application adaptation to highly varying conditions. To guide resource utilization decisions, applications should leverage network performance measurements including detailed statistics regarding connection status and data transfers.

In the BFT example, reducing the transmission rate and prioritizing the position information data flow would preserve system functionality and low latency despite a significant drop in available network resources.

Applications also benefit from notifications regarding disconnections so that they can adapt their behavior. With temporary disruptions, connections should enter a snooze state in which data transmission is suspended. Upon reestablishing connectivity, the reconnection procedure would be triggered. Applications could also take advantage of predictions on likely future channel conditions derived from trends in signal strength and signal to noise ratio.

Multimode Communications Support

Applications in tactical networks have multiple separate data flows with differing requirements. However, commonly used transport protocols such as TCP provide reliable and sequenced delivery of a single stream of data, which is inefficient over unreliable networks for three reasons. First, reliable delivery requires retransmission of lost or unacknowledged packets. Second, the stream-oriented abstraction prevents the transport protocol from being aware of independent application messages. Finally, sequenced delivery requires that data received out of order must be retained at the transport protocol level until ordering constraints have been satisfied. While these are useful abstractions, they are expensive to provide and, most important, are not needed by all applications at all times.

Therefore, the middleware should allow applications to exploit different message delivery semantics for each data flow, based on their needs. The communications middleware should provide flexible combinations of reliability, sequencing, and prioritization, and allow applications to specify their desired requirements. Providing this information to the transport protocol allows the middleware to better allocate scarce resources.

For instance, a BFT application might react to channel degradation by giving up reliability and/or sequencing, or imposing a constraint on the number of packet retransmissions, in order to obtain lower end-to-end latency.

Advanced Control of Transmission Queues

Data queuing deteriorates performance and increases latency. The middleware should provide applications with explicit functions to deal with time-sensitive information. An important approach to efficiency is to provide advanced control of transmission queues to reduce latency by minimizing unneeded message transmissions.

Data transmitted in tactical networks is often time sensitive (e.g., multimedia and control applications), which implies the data has a finite lifetime. The communications middleware should support specifying maximum lifetimes for outgoing messages so that outdated messages may be automatically discarded. Additionally, applications should be able to identify and discard specific messages from the transmission queue,

Tactical networks are characterized by limited bandwidth and intermittent connectivity. This suggests a fundamental requirement for application design: a continuous adaptation process to match both changing tactical objectives and network conditions, instead of assuming steady-state operations.

The middleware should provide applications with a platform-independent API. In addition, tactical networking middleware should be designed for easy deployment and tuning on legacy equipment and protocols commonly found in real-life tactical networks.

minimizing transmission of unnecessary data and significantly improving latency for delivery of necessary data.

For example, a BFT application might discard old position updates for an entity when a new one becomes available.

RUNTIME CONTROL OF RESOURCE ALLOCATION

Bandwidth scarcity in tactical network environments requires prioritization of channel access between competing applications in accordance with current tactical objectives. Access to network resources from low-priority or non-critical applications should be preempted in favor of critical applications. Therefore, the middleware must allocate network resources to competing demands to improve overall effective performance.

Tactical objectives and environmental conditions change continuously, causing frequent variations in application priorities for channel access. Tools and mechanisms to dynamically change resource allocations to applications at runtime are needed. A network administrator should be able to change the runtime behavior of distributed applications with an external policy-based control mechanism of runtime properties. This feature should enable the dynamic definition of policies to prioritize channel access for critical applications to provide mission performance that meets the commander's requirements. To support administrators with this task, the middleware should enable visualization of critical information about network and application performance over time.

In the BFT application, transmission of force positions from engagement zones and civilian zones are more important than force positions in quiescent zones.

INTEGRATION SUPPORT

The heterogeneity of applications and systems adopted in tactical networks requires portability to a wide range of hardware/operating system (OS) platforms, ease of integration, and backwards compatibility with legacy applications.

The middleware should provide applications with a platform-independent application programming interface (API). In addition, tactical networking middleware should be designed for easy deployment and tuning on legacy equipment and protocols commonly found in real-life tactical networks. Finally, the middleware should provide mechanisms that ease the migration of legacy and COTS applications to tactical networks, allowing their gradual porting on an as needed basis.

RELATED WORK

Many research efforts have addressed the problem of realizing efficient and robust communications in tactical networks and other highly dynamic wireless networking environments [9]. Several studies focus on improving TCP's robustness and performance, with the goal of preserving interoperability with the large base of existing TCP-based applications. Researchers have developed a plethora of different proposals, such as TCP-ELFN and TCP-BuS, which leverage cross-layer feedback mechanisms that provide notification of route or link failures from intermediate nodes along the communication path [10]. While these solutions achieve better performance than COTS TCP implementations, they do not provide applications with fundamental features for the tactical networking environment.

Other transport protocol proposals implement a richer programming model that goes beyond the reliable stream communication semantics of TCP. The Transport Protocol of the Space Communications Protocol Standards (SCPS-TP) [3] was originally developed for space communications, but is also proposed for tactical networks as these share common characteristics such as dynamic topology, intermittent connectivity, and bandwidth-constrained links with high bit error rate. SCPS-TP proposes a simple TCP-like API as well as an extended API with support for message-oriented communication, message prioritization, and full, partial, or minimal reliability. The Stream Control Transport Protocol (SCTP) [2] also supports message-based communications with partial reliability and unordered delivery, and allows several streams of communication in the same connection. While SCPS-TP and SCTP are better suited than TCP for tactical networks, they still fall short of providing applications with critical features such as traffic prioritization, advanced transmission queue control, and external control over runtime properties.

THE MOCKETS COMMUNICATION MIDDLEWARE

The Mockets middleware is an application-level communications middleware designed for tactical networks. Mockets addresses the specific requirements of this environment as detailed in the previous section by providing applications with a wider range of features and a richer programming model, which were purposely designed for tactical networks and evaluated in the context of several live experiments. This section briefly summarizes the important features of Mockets. Additional details may be found in [11, 12].

The Mockets research project began in 2003 with ARL's participation in QL-1. This experiment provided an opportunity to understand the types of applications used in tactical networks and their communication requirements. Moreover, observing the network and application behavior provided valuable insight into essential requirements and abstractions for the middleware. Participation in QL-2 in 2004 and C4ISR OTM from 2006 to 2008 facilitated an iterative design, testing, and evaluation process for the Mockets middleware. Lessons learned from each successive experiment led to enhancements to the design and implementation of Mockets in the context of ground-based tactical edge networks. Finally, in 2008, AFRL's participation in a live flight demonstration of a Joint Surveillance Target Attack Radar System (J-STARS) aircraft communicating with a portable ground station provided valuable insight into the airborne networking environment, allowing for the extension of Mockets to support this environment. During these live experiments, we had the opportunity to deploy Mockets on a wide variety of network platforms, ranging from 802.11-based ad hoc nodes to JTRS prototypes to currently deployed radios such as PSC-5.

MULTIMODE COMMUNICATIONS

Mockets supports and fosters a closed-loop feedback based adaptive programming model. Applications provide the middleware with messages and metadata on how to deliver them. In turn, the middleware provides applications with information about current network conditions so that they can tailor their current service rate and message delivery semantics.

More specifically, Mockets provides applications with connection-oriented message-based communication facilities. Mockets allows applications to create several data flows for each connection. Applications can assign specific message delivery semantics and QoS policies to each flow and change them at runtime. Specifically, Mockets allows applications to choose between orthogonal reliable/partially reliable[2]/unreliable and sequenced/unsequenced delivery semantics.

MESSAGE REPLACEMENT

Mockets provides advanced queue control functions that allow applications to prioritize the transmission of a specific message within one flow, to delete outdated messages in the transmission queue, or to replace them with newer messages. These functions were designed to reduce network traffic and message latency, and proved to be very effective when dealing with time-sensitive information such as position updates in BFT applications.

RICH APPLICATIONS PROGRAMMING INTERFACE

Mockets enables the development of adaptive applications by monitoring network status and providing the collected information. For convenience, Mockets offers a double interface: applications can either directly query the middleware or request to be notified via callbacks when a specific event occurs, such as the connection quality falling below a threshold.

Finally, Mockets can send connection status information, including event notifications such as connection setup, teardown, and connectivity loss, to an external monitoring application. This feature enabled development of the Mirage visualization tool (Fig. 2), which proved invaluable to monitor the runtime behavior of applications in live experiments and identify and correct anomalous behavior.

IMPLEMENTATION APPROACH

Implementing Mockets as middleware provided several advantages including phased integration and portability. Mockets uses UDP and hence operates on any platform supporting a TCP/IP stack, regardless of the underlying hardware and operating system. We have developed applications running on gateway nodes to use both Mockets and TCP concurrently — Mockets to communicate with tactical edge nodes and TCP to communicate with higher-echelon legacy systems. Mockets is implemented in C++, with bindings for Java and C#. Mockets also provides a TCP-compatible API that allows COTS applications to be gradually ported to the tactical environment, supporting and facilitating a phased transition process.

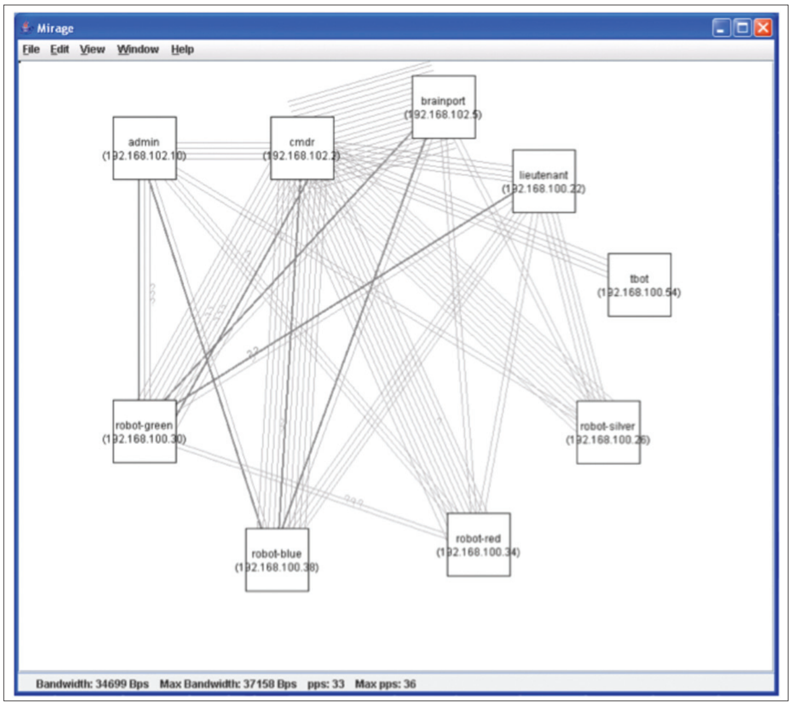
Mirage
File Edit View Window Help
admin (192.168.102.10)
cmdr (192.168.102.2)
brainport (192.168.102.5)
lieutenant (192.168.100.22)
tbot (192.168.100.54)
robot-green (192.168.100.30)
robot-silver (192.168.100.26)
robot-blue (192.168.100.38)
robot-red (192.168.100.34)
Bandwidth: 34699 Bps Max Bandwidth: 37158 Bps pps: 33 Max pps: 36

Figure 2. *Mirage visualizer showing active Mocket connections.*

EXPERIMENTAL RESULTS

We present results from two experiments to illustrate the performance advantages that can be achieved through an integrated and tailored approach like Mockets. The first results are from AFRL J-STARS ANE, which used an ARC-231 radio communicating with a ground-based PSC-5 tactical radio link. The PSC-5 adopts a demand-assigned multiple access (DAMA) MAC protocol on top of time-division multiple access (TDMA) with adaptive slot size and provides a half-duplex link with a maximum data rate of 56 kb/s.

During the experiment, the aircraft transmitted a series of JTIDS (ground track) messages in addition to servicing queries for generic text messages (ROE, ATO) from the ground client. In all, 113 JTIDS messages and 293 query results were received from the aircraft by the ground client.

As the data transfer was highly asymmetrical, DAMA assigned most of the communication channel to the ground station, effectively preempting channel use from the aircraft. This caused poor performance using COTS TCP implementations because of timeouts in ACK transmissions from the aircraft to the ground station. We achieved an 8× performance improvement by adapting Mockets to work over a half-duplex link. Specifically, we disabled congestion control and forced the middleware to periodically release the channel by not transmit-

[2] *Partial reliability is a trade-off between reliability at any cost (like TCP) and unreliable delivery (like UDP). The essence of partial reliability is constraining the resources expended for reliability — for example, in terms of limiting the number of retransmission attempts prior to discarding a message.*

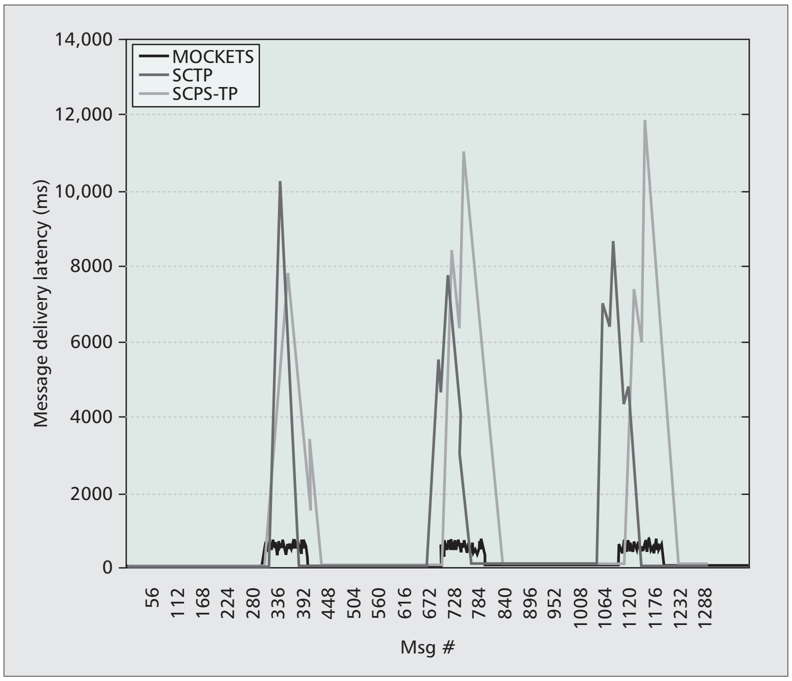

Figure 3. *Latency comparison between Mockets, SCTP, and SCPS-TP.*

ting for a few hundred milliseconds after a configured period of time.

We replicated the scenario in a controlled laboratory environment using a hardware radio emulator. The results collected in the laboratory, consistent with those achieved in the field, show that Mockets performs better than TCP, providing the same 8× improvement in throughput (3292 b/s for Mockets and 432 for TCP). Most other transport protocols use acknowledgment-based clocking and congestion control mechanisms similar to TCP and would therefore show similar results.

The second experiment demonstrates the effectiveness of Mockets-enabled advanced transmission queue control in mitigating the impact of abrupt network changes on latency. To this end, we evaluate the performance of a surrogate BFT application using Mockets, SCPS-TP, and SCTP.

The BFT application transmits 1 kbyte updates for each node at a rate of 1 Hz. With a seven-node configuration, clients receive updates at an average rate of one every 142 ms, for a total throughput of 7 kbytes/s. To reduce latency, we configured the Mockets-based application to exploit message replacement. More specifically, new position messages replaced previous messages in the transmission queue. This BFT application operated on an emulated network that provided a 30 kb/s link. Periodically (every 45 s), the quality of the link dropped to a 5 kb/s link for 15 s. Figure 3 shows the measured delivery latency for the first 1300 messages transmitted.

When the bandwidth drops, messages accumulate in the transmission queue. By replacing old enqueued messages when a new message becomes available, Mockets transmits fewer and more recent messages, thus achieving a 16–24× decrease in latency. The latency improvement is made possible by Mockets having additional information from the application on how to handle the data, delineating the traffic into individual messages, and having sufficient information to determine when a new message makes previously enqueued messages obsolete.

CONCLUSIONS

In our experience with net-centric operations, the peculiar characteristics of tactical networks present challenging issues that need to be specifically addressed in the realization of net-centric applications. Our work demonstrates that a communication middleware for tactical environments specifically designed to consider application requirements and communication patterns can significantly improve both performance and robustness. The Mockets communications middleware addresses many of these challenges, as evidenced by our experiments in the field and in laboratory settings.

We hope that readers will benefit from our experiences and understand the importance of tailoring the design and implementation of transport protocols and communication systems to both the applications and the target environments. Enhanced APIs that allow better integration between applications and the middleware are important. These APIs need to support a phased transition path from COTS protocols, such as TCP, that are designed for Internet-style environments. We hope that our experiences and results also lead to future standardization efforts for middleware and transport protocols suitable for tactical environments.

ACKNOWLEDGMENTS

This research was sponsored in part by the U.S. Army Research Laboratory under Cooperative Agreement W911NF-04-2-0013, by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0009, and by the Air Force Research Laboratory under Cooperative Agreement FA8750-06-2-0064. Figure 1 is courtesy of the Northrop-Grumman Corporation, Melbourne, FL.

REFERENCES

[1] D. S. Alberts, J. J. Garstka, and F. P. Stein, "Network Centric Warfare: Developing and Leveraging Information Superiority," *CCRP Publication Series*, 2nd ed. (Revised), Aug. 1999; http://www.dodccrp.org/files/Alberts_NCW.pdf
[2] R. Stewart (Ed.), "Stream Control Transmission Protocol," RFC 4960, Sept. 2007.
[3] CCSDS Secretariat — NASA, "Space Communications Protocol Specification (SCPS) Transport Protocol (SCPS-TP) — Blue Book," Washington, DC, May 1999.
[4] R. Carl *et al.*, "Transport Protocols in the Tactical Network Environment," *Proc. IEEE Aerospace Conf. '07*, Mar. 3–10, 2007, pp. 1–9.
[5] R. Schmitz *et al.*, "The Impact of Wireless Radio Fluctuations on Ad Hoc Network Performance," *Proc. 29th Annual IEEE Int'l. Conf. Local Comp. Net.*, 2004, pp. 594–601.
[6] J. L. Burbank *et al.*, "Key Challenges of Military Tactical Networking and the Elusive Promise of MANET Technology," *IEEE Commun. Mag.*, vol. 44, no. 11, Nov. 2006, pp. 39–45.
[7] S. Schütz *et al.*, "Protocol Enhancements for Intermittently Connected Hosts," *ACM Comp. Commun. Rev.*, vol. 35, no. 3, July 2005, pp. 5–18.

[8] K. Lund *et al.*, "Using web services to Realize Service Oriented Architecture in Military Communication Networks," *IEEE Commun. Mag.*, vol. 45, no. 10, Oct. 2007, pp. 47–53.

[9] M. Carter, "A Review of Transport Protocols as Candidates for use in a Tactical Environment," Defense Science and Technology Organization Tech. Rep. DSTO-TR-1808, Australian Government Dept. of Defense, 2005.

[10] R. Jurdak, *Wireless Ad Hoc and Sensor Networks: A Cross-Layer Design Perspective*, Springer, 2007.

[11] N. Suri *et al.*, "Mockets: A Comprehensive Application-Level Communications Library," *Proc. IEEE MILCOM '05*, Atlantic City, NJ, Oct. 2005.

[12] M. Tortonesi *et al.*, "Mockets: A Novel Message-Oriented Communication Middleware for the Wireless Internet," *Proc. WINSYS '06*, Setúbal, Portugal, Aug. 2006.

BIOGRAPHIES

NIRANJAN SURI [M] (nsuri@ihmc.us) is a research scientist at the Florida Institute for Human and Machine Cognition (IHMC). He received his Ph.D. in computer science from Lancaster University, England, and his M.Sc. and B.Sc. in computer Science from the University of West Florida, Pensacola, Florida. His current research activity is focused on the notion of agile computing, which supports the opportunistic discovery and exploitation of resources in highly dynamic network environments. His other research interests include distributed systems, networking, communication protocols, virtual machines, and software agents. He has been a principal investigator of numerous research projects sponsored by the U.S. ARL, U.S. AFRL, Defense Advanced Research Projects Agency (DARPA), Office of Naval Research (ONR), and National Science Foundation (NSF). He has authored or co-authored over 50 papers, has been on the technical program committees of several international conferences, and has been a reviewer for NSF as well as several international journals.

MAURO TORTONESI (mauro.tortonesi@unife.it) received his Laurea degree in electronic engineering and his Ph.D. in computer science engineering from the University of Ferrara, Italy. From 2004 to 2005 he was a research associate at the Florida Institute for Human and Machine Cognition. He currently holds a post-doctoral research assistant position at the Engineering Department of the University of Ferrara, Italy. His research interests include distributed and mobile computing, QoS management, business-driven IT management, and industrial automation.

ERIKA BENVEGNU (ebenvegnu@ihmc.us) received her Bachelor's degree in computer science engineering from the University of Padova, Italy, in 2005. She continued her studies at the University of Ferrara, where she gained a Master's degree in computer science and automation engineering in 2008. She is currently a research associate at the Florida Institute for Human and Machine Cognition. Her research focuses on communications in mobile ad hoc networks as well as distributed control of multiple autonomous systems with emphasis on sensor fusion and coordination activities using robots.

CESARE STEFANELLI [M] (cesare.stefanelli@unife.it) received his Laurea degree in electronic engineering and his Ph.D. in computer science engineering from the University of Bologna, Italy. He is currently a professor of computer science engineering in the Engineering Department of the University of Ferrara. His research interests include distributed and mobile computing, adaptive and distributed multimedia systems, network and systems management, and network security.

JESSE KOVACH (jkovach@arl.army.mil) is a computer engineer with the Battlefield Information Processing Branch of the U.S. ARL, specializing in information dissemination in tactical networks and sensor and robotic systems integration. He designs, develops, and tests prototype systems that incorporate advanced concepts developed within the laboratory as well as elsewhere in the defense R&D community. His work has been incorporated into a number of fielded systems and has been used in multiple high-visibility demonstrations including C4ISR On the Move and Empire Challenge. He holds a Bachelor's degree in computer engineering from the University of Maryland.

JAMES HANNA (james.hanna@rl.af.mil) is a senior research engineer with the Enterprise Information Management Branch in the Information Directorate at the U.S. AFRL. He has been a computer engineer for AFRL for the past 20 years. He has over 22 years of experience developing software applications. For the past four years he has been concentrating his expertise in the area of network-centric tactical information management. He is currently the lead engineer for the Agile Information Environment research area. He is also the lead in-house engineer researching issues related to distributed information management, distributed policy enforcement, and advanced architectures to address the complex challenges of deploying information management in forward tactical environments. He has a B.Sc. in computer science from the State University of New York Institute of Technology, Utica. He was the 2004 AFRL Ralph I. Cole Engineer of the Year Award recipient.