ESRO: A Foundation for the Development of Efficient Protocols

Mohsen Banan

http://mohsen.banan.1.byname.net/ContactMe

Version 1.3
First Published: August 4, 2000
Last Updated: August 9, 2000

A Component of *The LEAP Manifesto*

This article is one of a series of articles describing various aspects of the Mobile Messaging industry and the Lightweight & Efficient Application Protocols (LEAP) protocols. For the complete collection of articles see *The LEAP Manifesto* [15], available at `http://www.LeapForum.org/LEAP/Manifesto/roadMap/index.html`. *The LEAP Manifesto* is also available at the Free Protocols Foundation website at `http://www.FreeProtocols.org/LEAP/Manifesto/roadMap/index.html`.

Contents

List of Figures

List of Tables

1 Overview of ESRO

All efficient applications have the requirement for an efficient transport mechanism. For this reason, part of the initial focus of the LEAP protocol development effort has been on creating a general efficient transport mechanism. The resulting protocol is referred to as **Efficient Short Remote Operations**, or **ESRO**. ESRO is the efficient transport layer protocol for several LEAP applications. ESRO is a reliable connectionless transport mechanism, forming the foundation for the development of efficient protocols when TCP is too much and UDP is too little.

The need for efficient protocols extends across all aspects of wireless data communications, including e-mail, web browsing, and other applications. The LEAP architecture accommodates many of these applications based on the use of ESRO.

ESRO was published as RFC-2188 [3] in September of 1997. The ESRO protocol is publicly maintained and enhanced by ESRO.org at http://www.esro.org/. Patent free declarations have been made with respect to ESRO through the Free Protocols Foundation.

1.1 The Need for ESRO

Considering that:

- Many vertical applications in the wireless environment need a *reliable*, *efficient* and *lightweight* data transfer mechanism.
- Existing Internet transport protocols (TCP/UDP) do not address these requirements adequately.
- Today, most vertical applications either implement their own *ad hoc* reliability service on top of UDP, or else use non-standard middleware.
- A reliable connectionless transport protocol based on open specifications is needed.
- Remote Operations is superior to reliable connectionless transport as an application development paradigm.

ESRO has been designed to address these specific needs.

The need to address the requirements that the ESRO protocol addresses has been well recognized for quite a long time. As early as 1995 such requirements were being discussed within the Internet research community. In particular, RFC-955 [6], entitled “Towards a Transport Service for Transaction Processing Applications,” demonstrates recognition of the need for ESRO. A more recent document, RFC-2757 [17], entitled “Long Thin Networks,” again recognizes the demand for such a protocol.

In the past, this unaddressed demand has given rise to a variety of product oriented proprietary solutions (as opposed to open protocols) which are often referred to as “Middleware Products.” This has been particularly widespread in the wireless industry. Often these solutions are vendor specific and do not scale.

1.2 ESRO Requirements and Goals

The requirements and goals driving the ESRO protocol and system design include the following:

1. Provide reliability in an efficient manner for a wide range of vertical applications (e.g. wireless).
2. Specify an Internet open protocol.
3. Minimize the number of transmissions.

4. Minimize the number of bytes transmitted.
5. Be fast; minimize latency.
6. Be power efficient, and show respect for the resource limitations of mobile platforms, including memory, CPU, and battery power limitations.
7. Be lightweight; accommodate miniaturized devices.

1.3 Terminology

A description of ESRO includes references to a number of basic data communications concepts and ideas. Because of the informal, *ad hoc*, and often inconsistent terminology used within the Internet technical community, many of these concepts are referred to by a variety of different names throughout various documents and RFCs. Many of the RFCs mentioned below refer to ESRO-related concepts in an inconsistent manner.

In contrast to this, the ESRO specification uses the well defined and precise terminology of the "Open Systems Interconnection Reference Model" [18]. In this paper we will also adhere to the same precise terminology.

An example of the use of imprecise terminology is the term "Transaction Processing." Various other protocol specifications refer to the concept of "Remote Operations" as "Transaction Processing." In our terminology, however, Transaction Processing goes above and beyond Remote Operations, and also includes the concepts of Commitment, Concurrency and Recovery, and chained (related) Remote Operations which may be built on top of ESRO.

2 Other Related Protocols

The overall model of ESRO is similar to and consistent with many existing protocols. The distinguishing characteristic of ESRO is its efficiency. Also, the scope of ESRO is very narrow and well defined. The options and selections that it provides for are few and clear.

A brief comparison of ESRO to other similar protocols is provided in the sections below.

2.1 RPC

Remote Procedure Call (RPC) is specified in RFC-1831 [23] and RFC-1833 [22].

The RPC specifications define a remote procedure model that is essentially the same as ESRO. However, the notation of RPC uses a syntax which is quite different from that of ESRO. RPC can rely on a connection-oriented or a connectionless transport mechanism. When using the connectionless mechanism, the retransmission and reliability issues are considered to be beyond the scope of the RPC specification. RPC is usually used in combination with External Data Representation, XDR (RFC-1832) [24].

When using RPC over UDP, no meaningful reliable transport mechanism is defined. For this reason use of RPC over UDP has been limited to LANs.

2.2 ROSE

Remote Operations Services Element (ROSE) is specified in [7] and [8]. ROSE is a complete and heavyweight traditional OSI application which assumes availability of all of the underlying OSI layers.

The ESRO protocols can accomplish short operations with much less overhead than ROSE.

2.3 WAP's WTP

The Wireless Appliction Protocol (WAP) includes a layer which is similar to ESRO, called "Wireless Transaction Protocol" (WTP) [1].

The WTP specification was published after ESRO was published, and is similar in functionality to ESRO. In many ways WTP can be considered to be patterned after ESRO; however, WTP is in fact a step backwards.

The clear and simple Remote Operations model of ESRO is renamed as "Wireless Transactions" in an inappropriate context. The notation specification conventions are not used, and the state machines are not as robust.

More importantly, the WTP specifications are not stable, have not been published as Internet RFCs, and have not been declared to be patent free.

As early as 1995, those involved in the development of WTP were made aware of LSROS and ESRO. The only reason we know of for their re-specification of WTP, rather than re-use of ESRO, is the WAP Forum's desire for control.

More details on the problems of the WAP Forum's approach are provided in the article *The WAP Trap* [16].

2.4 T/TCP

Transaction/TCP is specified in [4] and [5]. T/TCP is a backwards-compatible extension of TCP which provides efficient transaction-oriented service in addition to virtual-circuit service.

Use of T/TCP often involves replacing existing TCP implementations. This non-evolutionary aspect of T/TCP is one of the reasons why T/TCP has not been widely adopted.

Various lessons can be learned from why T/TCP has not been widely adopted.

2.5 RDP

Reliable Data Protocol (RDP) is specified in [25] and [19].

RDP can be considered to be a specialized TCP, specified for particular circumstances in which some of TCP's facilities are needed.

One of the reasons why RDP has not been heavily used is because the set of specialized circumstances that it addresses do not justify the cost of implementing it. RDP allows for many options and selections, which makes its use difficult.

Various lessons can be learned from why RDP has not been widely adopted.

2.6 VMTP

Versatile Message Transaction Protocol (VMTP) is specified in [9].

VMTP can be considered to be a specialized transport protocol, targeted for what it calls the transaction model of communications.

One of the reasons why VMTP has not been heavily used is because it tries to address too broad of a software engineering-centric model. VMTP allows for many options and selections, which makes its use difficult.

Various lessons can be learned from why VMTP has not been widely adopted.

2.7 TCP

Transmission Control Protocol (TCP) is specified in [21] and [5].

TCP is mature, well understood and stable. Congenstion control and flow control mechanisms for TCP have been developed over the years, and incorporated into TCP implementations.

The simplest and shortest TCP interaction involves 5 PDU exchanges. For applications wishing to accomplish short and occasional communications in less than 5 PDU exchanges, ESRO is more efficient that TCP.

2.8 UDP

UDP (User Datagram Protocol) is specified in [20].

UDP is a very simple and thin layer on top of IP, which does not provide reliability or sequence ordering. Applications requiring a reliable connectionless transport need to build on top of UDP. ESRO provide an efficient and reliable transport service on top of UDP.

2.9 UDP Plus Ad Hoc Re-Transmissions

Various protocols have added their own specific re-transmission policies on top of UDP to make it more reliable.

Examples of such efforts include the Domain Name System (DNS) [11] [12], Network Time Protocol (NTP) [10], and NNTP for Usenet News Reading.

These efforts have all resulted in incomplete and limited solutions. The problems with such approaches were understood as early as 1985 [6].

3 The ESRO Protocol

The ESRO specification describes the service model, the notation and the protocol for Efficient Short Remote Operations (ESRO). The ESRO service is similar to and is consistent with other Remote Operation and Remote Procedure Call services. The emphasis of the ESRO service definition and the ESRO protocol is on efficiency. ESRO is designed specifically with wireless network (e.g. CDPD) usage in mind.

The ESRO protocol provides reliable connectionless remote operation services on top of UDP (or any other non-reliable connectionless transport service) with minimum overhead. ESRO supports segmentation and reassembly, concatenation and separation, as well as multiplexing for service users (applications).

ESRO allows for trade-offs between efficiency and reliability by specifying both 2-way handshake and 3-way handshake based protocols.

The ESRO specifications also define a notation and the services provided by an application-service element to support interactive applications in a distributed systems environment. A *Remote Operation* is invoked by one entity; the other entity attempts to perform the Remote Operation and then reports the outcome of the attempt.

Encoding mechanisms for presentation of the parameters of remote operations are outside the scope of ESRO. However, identification (tagging) of the encoding mechanism in use (e.g. XDR, BER, PER) is supported by ESRO.

A variety of applications can use the ESRO protocol. Some early applications which use ESRO include: efficient short message submission and delivery, credit card authorization, and white pages lookup.

3.1 Efficiency Characteristics of ESRO

The key requirement driving the design of ESRO is efficiency. Reliable transfer of a short message using TCP involves 5 transmissions at a minimum. Reliable transfer of a short message using ESRO involves only 3 or 2 transmissions.

For many applications in which optimization of efficency is desired, it is likely that elimination of the extra 2 transmissions which are inherent to TCP, justifies deviation from it and use of ESRO instead.

The efficiency premium realized by the use of ESRO rather than TCP can be very significant. For example, EMSD (a mail submission and delivery protocol that uses ESRO) can be upto 5 times more efficient than SMTP, while maintaining precisely the same level of reliability and security. The paper entitled “Efficiency Study of EMSD vs. SMTP/POP3/IMAP” [13] quantifies the efficiency of ESRO in comparison to traditional TCP based applications.

3.2 Why We Adopted the Remote Operations Model

ESRO is a reliable connectionless transport mechanism.

A reasonable question is: Why did we design ESRO’s service interface to be based on the Remote Operations model?

Many Internet protocols are ”text-based” on top of TCP. And the “here is some text” followed by “here is some more text” followed by “here is some text in response” has become the tradition of simple Internet protocols. Protocols designed on the basis of this ”text-based” approach have a good track record of acceptance throughout the Internet, primarily because they are simple to understand and simple to implement.

When efficiency matters, however, the traditional text exchange model can be better expressed by the client requesting a particular operation from the server, and the server responding with the results of that operation, thereby eliminating the traditional “text-based” chit-chat. With such an approach, the design of the protocol becomes a natural fit for the remote operations model.

For short interactions, a reliable connectionless transport mechanism and the Remote Operations model are simply the same thing. The formalism of Remote Operations is an asset that ESRO exploits.

ESRO provides a service which supports interaction of applications based on a remote operation model. A Remote Operation is invoked by one entity; the other entity attempts to perform the Remote Operation and then reports the outcome of the attempt. The ESRO protocol is designed to be able to support a large variety of applications.

3.3 RFC Publication of the ESRO Protocol

The ESRO protocol is completely open. It has been published as RFC-2188.

For information on how to obtain the ESRO protocol, visit the ”Base Protocol Specifications” section of http://www.esro.org/.

3.4 Maintenance of the ESRO Protocol via ESRO.org

ESRO.org is a public organization which enables and facilitates the development, maintenance and enhancement of protocols and related technologies which address the efficiency requirements of generic Internet applications.

Anyone interested in participating in the development of the ESRO protocol can join ESRO.org by visiting the ”Joining the ESRO.org and Related Mailing Lists” section of http://www.esro.org/.

Patent-free declarations have been made with respect to ESRO and RFC-2188 through the Free Protocols Foundation.

4 Use of ESRO

The ESRO protocol is designed to be able to support many applications, such as mobile messaging, directory services, micro-browsers, dictionary lookup (white and yellow pages), data sensors monitoring, telemetry, dispatch, and credit card authorization applications.

In this section we discuss various considerations that protocol developers and application designers should make when designing applications which use ESRO.

We then build on this by providing various examples.

4.1 Common ESRO Application Design Considerations

When designing applications which use ESRO services, a number of common issues often need to be considered. These include:

- Presentation – Syntax and Encoding
- Operation Reliability
- Idempotency – Duplication Detection
- Failure Detection and Re-Tries
- Segmentation/Re-Assembly
- Congestion and Flow Control
- Security Considerations: Authentication, Confidentiality, Authorization

We discuss each of these in some detail below.

4.1.1 Presentation – Syntax and Encoding

Using the remote operations model, Argument, Result and Error parameters are communicated between the invoker and the performer.

The application designer must choose and specify the particular syntax and encoding to be used.

Encoding mechanisms for presentation of the parameters of remote operations are outside the scope of ESRO. However, identification (tagging) of the encoding mechanism in use is supported by ESRO.

The choice of encoding mechanism by the developers often revolves around the efficiency requirements, the complexity of the application to be designed, the availability of tools (e.g. XML, XDR, ASN parser and compilers) and the familiarity of the developer with the tools.

ESRO can accomodate any syntax and encoding, including: plain text, XML, ASN.1 (BER, PER etc.) and XDR.

The ESRO specification introduces one notation in ASN.1 for representation of Operations. This in no way ties the use of ESRO to ASN.1. Any syntax and encoding can be used, and use of any notation for representation of Operations is not mandatory.

4.1.2 Operation Reliability

Different applications may need various levels of failure detection for various operations. Any one of the following three methods may be needed to accomplish the desired reliability and semantics of various aspects of applications:

- 2-Way Handshake
- 3-Way Handshake
- 3-Way Handshake with Verify

An example of the usage of all three of the above methods is provided by EMSD [2].

4.1.3 Idempotency – Duplication Detection

Some operations are idempotent in nature, that is, they can be performed more than once without causing harm. However, other operations are non-idempotent, and should therefore be performed only once. In the case of non-idempotent operations, the performer should be able to detect duplicate operations, and perform each non-idempotent operation only once.

Examples of non-idempotent operations are Submission and Delivery of messages, which should not be performed more than once. Examples of idempotent operations are enquiries of date and time, which can be performed more than once without harm.

ESRO Services do not detect duplicate invocation of operations. As a result, the application should detect duplication when the same operation instance is invoked more than once.

An example of how this can be accomplished in a coherent manner can be found in Section 4 of RFC-2524, entitled "Duplicate Operation Detection Support" [2].

4.1.4 Failure Detection and Re-Tries

Based on ESRO's notification of Failures, the application must take necessary measures to re-try the operation.

Depending on the nature of the application, use of a general spooler which supports persistence may be reasonable.

4.1.5 Segmentation/Re-Assembly

ESRO provides Segmentation and Re-Assembly, as well as Concatenation and Separation, inside the protocol. However, an application may choose to provide segmentation and re-assembly above ESRO services.

The specific requirements of the application and the nature of the networks in use may justify such a design.

4.1.6 Congestion Control and Flow Control

The ESRO protocol includes flow control, and allows for various re-transmission timers policies to be implemented. Such policies may include exponential back-off and adaptive retransmission algorithms. These allow for the use of self-adjusting timers to determine and dynamically set timers to effectively adjust data traffic in the event the link is slower than usual due to congestion or other network or system conditions.

However, specification of retransmission timers policies was deliberately not included in the ESRO protocol. Often ESRO will be used on the edges of the Internet, where the characteristics of network behavior between the Invoker and the Performer are deterministic and stable. In such environments, a custom re-transmission timers policies is more effective than a generic one. For example, timer profiles specific to CDPD or GSM wide area networks which assume the servers are at the edges of the Internet can be tailored to optimize performance. In practice, early usages of ESRO are likely to be in such environments.

Congestion Control was also deliberately not included in the ESRO protocol, and is intended to sit on top of ESRO. This is for two reasons.

First, indications of congestion in the IP protocol suite violate layering principles, and use of such indications would have made the congestion control mechanism limited to IP. Use of ESRO over non-IP packet networks (such as GSM) is highly desirable in the short term.

Second, the indication and source congestion need not be limited to purely network resources, and indications of congestion may come from a variety of sources. In practice, ESRO based congestion control is primarily server driven. Design of ESRO applications should include Congestion Control mechanisms.

An example of how Congestion Control on top of ESRO may be designed can be found in SubmissionControl and DeliveryControl sections of RFC-2524, [2]. Such control facilities and design makes use of ESRO and EMSD well suited for the End-To-End Internet usage model.

4.1.7 Security

Different applications may need various levels of security facilities. It is the requirements and scope of the individual application that drives the nature of security facilities that are appropriate to use.

In many cases, the key required security facilities are Authentication, Confidentiality and Authorization. The trade-offs between complexity, efficiency and reliability are best made on a case-by-case basis.

In the specific case of Efficient Mail Submission & Delivery (EMSD) [2], the existing mainstream Internet security mechanism of Pretty Good Privacy (PGP) can be used to provide end-to-end security facilities.

As an underlying general facility, ESRO itself does not provide any generalized security facilities. At some point in time, it makes good sense to create a common security framework for LEAP which is consistent with the broader Internet security framework.

We will start incorporating common security features when they become widespread in the broader Internet. When Network Layer Security, Transport Layer Security, PKI, etc. are widespread, they will be incorporated as common facilities within LEAP.

5 Example Applications

A variety of applications can make use of the ESRO protocol. In this section we mention and provide references to a number of ESRO based applications. Based on the broadness of their usage and intended audience, we categorize them as either “Horizontal Applications” or “Vertical Applications.”

5.1 Horizontal Applications

Various efficient horizontal applications have been developed or are being developed using ESRO. These include:

- EMSD: Mobile and Wireless Messaging, Modern Two-Way Paging
- EHTD: Efficient Micro-Browsing
- E-DICT: Efficient Dictionary Lookup

A brief description of each is provided below.

5.2 EMSD: Efficient E-Mail

One of the efficient application layers built on top of ESRO is called Efficient Mail Submission & Delivery, or EMSD. EMSD is the component of LEAP that addresses the Mobile Messaging application.

EMSD is highly optimized for the submission and delivery of short (typically 4 kilobytes or less) Internet e-mail messages, and is therefore extremely well suited to the wireless environment. EMSD improves on existing messaging protocols by optimizing the exchange between the server and the end-user device, both in terms of the number of bytes transferred, and in terms of the number of transmissions. For more information on EMSD see the article *EMSD: The LEAP E-Mail Component* within *The LEAP Manifesto*, or visit the EMSD website at http://www.emsd.org/.

5.3 EHTD: Efficient Web Browsing

The Efficient Hyper Text Delivery (EHTD) layer is a hypertext transfer protocol which is optimized for the efficient transfer of short markup pages. EHTD is the component of LEAP which facilitates web browsing. Along with EMSD, EHTD also benefits from the reliable efficient services of ESRO. A multiplicity of efficient markup languages can be used in conjunction with EHTD. Development of the EHTD protocol is currently in progress.

5.4 Other Efficient Horizontal Applications

Various other efficient application protocols are either under development, or anticipated for future development. One of these is the Efficient Dictionary protocol, or E-DICT, which will enable efficient access to dictionaries and other lookup data structures. A starting point for the E-DICT protocol is currently being created. In developing E-DICT, we intend to build on the existing work already done in the context of the DICT protocol.

We anticipate that additional protocols will be needed for a variety of future applications, not all of which can be foreseen at this time. These applications will include such things as efficient implementations of ESRO-based instant messaging, chat, white pages, and others.

5.5 Vertical Applications

Various efficient vertical applications have been developed or are being developed using ESRO. These applications include:

- Wireless credit card verification
- Wireless automated teller machine (ATM)
- Dispatch applications
- Telemetry
- Wireless T.V. ratings
- Vehicle tracking (GPS applications etc.)

There are a set of common architectural characteristics to most of these vertical applications. These common characteristics are illustrated in Figure 1. In this figure, the box labeled “Thin Reliability Layer” represents the position of ESRO.

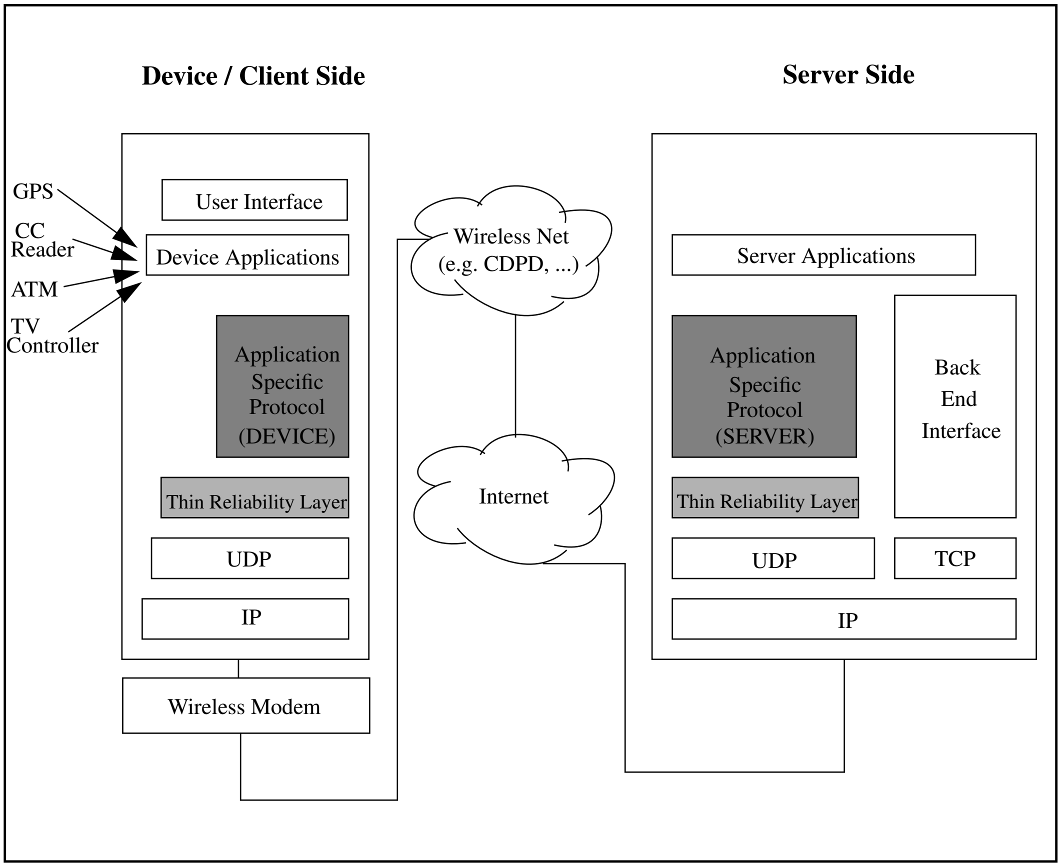

Figure 1: Anatomy of a Vertical Wireless Application

The combination of ESRO as the efficient transport mechanism, the systems modeling represented in Figure 1, and the methods described in Section 4.1 can greatly facilitate and expedite development of efficient vertical applications.

6 Existing Implementations of ESRO

A Reference Implementation of ESRO in the form of free software in open-source format is being made available at http://www.MailMeAnywhere.org/.

We expect to have the ESRO protocol engine software components subject to the GNU General Public License available at this location by September 2000.

On the device side, software will be made available for most generic platforms, including PalmOS, EPOC, WindowsCE, Windows9X, and Windows-NT. On the server side, software will be available on Solaris, Linux and NT.

In addition to open-source format, the ESRO Reference Implementation will also be made available for the above mentioned platforms as a ready-to-use Toolkit.

For a current list of ESRO free and commercial software products, visit the ”Products and Services” section of http://www.esro.org.

6.1 ESROS Application Programming Interface

The "ESRO API Specification" [14] defines the ESRO Application Programming Interface (API). This specification maps ESRO service primitives into C language function calls. This document is available at http://www.esro.org/documents/API.html.

References

[1] *WAP Wireless Transaction Protocol Specification*, February 2000. The WAP-201-WTP article available at http://www.wapforum.org/what/technical.htm.

[2] M. Banan. Neda's Efficient Mail Submission and Delivery (EMSD) Protocol Specification Version 1.3. RFC 2524 (Informational), February 1999.

[3] M. Banan, M. Taylor, and J. Cheng. AT&T/Neda's Efficient Short Remote Operations (ESRO) Protocol Specification Version 1.2. RFC 2188 (Informational), September 1997.

[4] R. Braden. Extending TCP for Transactions – Concepts. RFC 1379 (Informational), November 1992. Updated by RFC 1644.

[5] R. Braden. T/TCP – TCP Extensions for Transactions Functional Specification. RFC 1644 (Experimental), July 1994.

[6] R.T. Braden. Towards a transport service for transaction processing applications. RFC 955, September 1985.

[7] Remote Operations: Model, Notation and Service Definition. The International Telegraph and Telephone Consultative Committee, March 1988. Recommendation X.219.

[8] Remote Operations: Protocol Specification. The International Telegraph and Telephone Consultative Committee, March 1988. Recommendation X.229.

[9] D.R. Cheriton. VMTP: Versatile Message Transaction Protocol: Protocol specification. RFC 1045 (Experimental), February 1988.

[10] D. Mills. Network Time Protocol (Version 3) Specification, Implementation and Analysis. RFC 1305 (Draft Standard), March 1992.

[11] P.V. Mockapetris. Domain names - concepts and facilities. RFC 1034 (Standard), November 1987. Updated by RFCs 1101, 1183, 1348, 1876, 1982, 2065, 2181, 2308, 2535, 4033, 4034, 4035, 4343, 4035, 4592.

[12] P.V. Mockapetris. Domain names - implementation and specification. RFC 1035 (Standard), November 1987. Updated by RFCs 1101, 1183, 1348, 1876, 1982, 1995, 1996, 2065, 2136, 2181, 2137, 2308, 2535, 2845, 3425, 3658, 4033, 4034, 4035, 4343.

[13] Mohsen Banan. *Efficiency Study of EMSD vs. SMTP/POP3/IMAP*. Neda Published Document 103-101-01.01, EMSD Organization, 1996. Online document is available at http://www.emsd.org/pubs/biblio/103-101-01-01/index.html.

[14] Mohsen Banan. *ESROS Application Programming Interface*. Neda Published Document 103-101-06.03, ESRO Organization, 1996. Online document is available at http://www.esro.org/pubs/biblio/103-101-06-03/index.html.

[15] Mohsen Banan. *Lightweight & Efficient Application Protocol (LEAP) Manifesto*. Technical Report 108-101-01, LEAP Forum, Bellevue, WA, January 2000. Online document is available at http://www.leapforum.org/LEAP/Manifesto/completeManifesto.

[16] Mohsen Banan. *The WAP Trap*. FPF Published Document 108-102-01, Free Protocols Foundation, Bellevue, WA, January 2000. Online document is available at http://www.freeprotocols.org/wapTrap.

[17] G. Montenegro, S. Dawkins, M. Kojo, V. Magret, and N. Vaidya. Long Thin Networks. RFC 2757 (Informational), January 2000.

[18] Information Processing Systems — Open Systems Interconnection: Basic Reference Model. International Organization for Standardization and International Electrotechnical Committee, 1984. Interational Standard 7498.

[19] C. Partridge and R.M. Hinden. Version 2 of the Reliable Data Protocol (RDP). RFC 1151 (Experimental), April 1990.

[20] J. Postel. User Datagram Protocol. RFC 768 (Standard), August 1980.

[21] J. Postel. Transmission Control Protocol. RFC 793 (Standard), September 1981. Updated by RFCs 1122, 3168.

[22] R. Srinivasan. Binding Protocols for ONC RPC Version 2. RFC 1833 (Proposed Standard), August 1995.

[23] R. Srinivasan. RPC: Remote Procedure Call Protocol Specification Version 2. RFC 1831 (Proposed Standard), August 1995. Obsoleted by RFC 5531.

[24] R. Srinivasan. XDR: External Data Representation Standard. RFC 1832 (Draft Standard), August 1995. Obsoleted by RFC 4506.

[25] D. Velten, R.M. Hinden, and J. Sax. Reliable Data Protocol. RFC 908 (Experimental), July 1984. Updated by RFC 1151.