University of Glasgow

Reliability and Data Transfer

Networked Systems (H)

Lecture 5

Packet Loss in the Internet

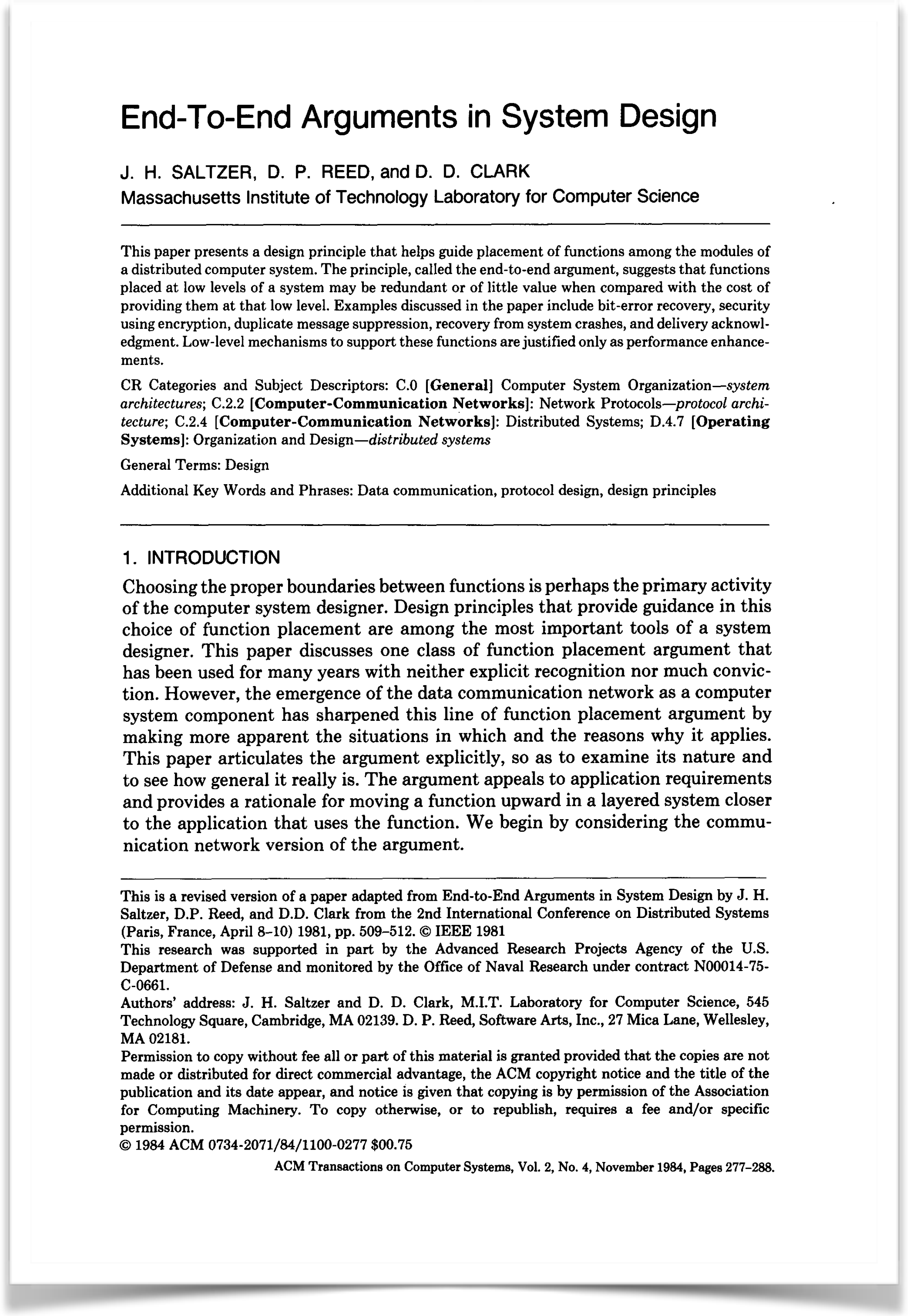
- Best effort traffic
- The end-to-end argument
- Timeliness vs. reliability trade-off

Packet Loss in the Internet

- The Internet is a best effort packet delivery network – **it is unreliable**
  - IP packets may be lost, delayed, reordered, or corrupted in transit
  - How often this happens varies significantly
    - Wireless links are less reliable than wired links
    - Countries with well-developed infrastructure tend to have reliable Internet links; countries with less robust or lower capacity infrastructure tend to see more problems
    - Some protocols intentionally try to push links to capacity, causing temporary overload as they try to find the limit
      - TCP and QUIC do this, when certain widely congestion control algorithms are used → lecture 6
    - **Some packet loss is inevitable**
- The **transport layer** must adapt the quality of service provided by the network to match application needs

The End-to-End Argument

- Is it better to place functionality within the network or at the end points?
  - Only put functionality that is absolutely necessary in the network, leave everything else to end systems
    - **Example**: let the network provide best effort packet delivery, rather than try to detect and retransmit lost packets
    - If the network is not guaranteed to be **100%** reliable, always, end systems must check for lost packets anyway
    - Since 100% reliability can never be guaranteed, no point in complicating the network trying to make it reliable
- One of the defining principles of the Internet



End-To-End Arguments in System Design

J. H. SALTZER, D. P. REED, and D. D. CLARK
Massachusetts Institute of Technology Laboratory for Computer Science

This paper presents a design principle that helps guide placement of functions among the modules of a distributed computer system. The principle, called the end-to-end argument, suggests that functions placed at low levels of a system may be redundant or of little value when compared with the cost of providing them at that low level. Examples discussed in the paper include bit-error recovery, security using encryption, duplicate message suppression, recovery from system crashes, and delivery acknowledgment. Low-level mechanisms to support these functions are justified only as performance enhancements.

CR Categories and Subject Descriptors: C.0 [**General**] Computer System Organization—*system architectures*; C.2.2 [**Computer-Communication Networks**]: Network Protocols—*protocol architecture*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems; D.4.7 [**Operating Systems**]: Organization and Design—*distributed systems*

General Terms: Design

Additional Key Words and Phrases: Data communication, protocol design, design principles

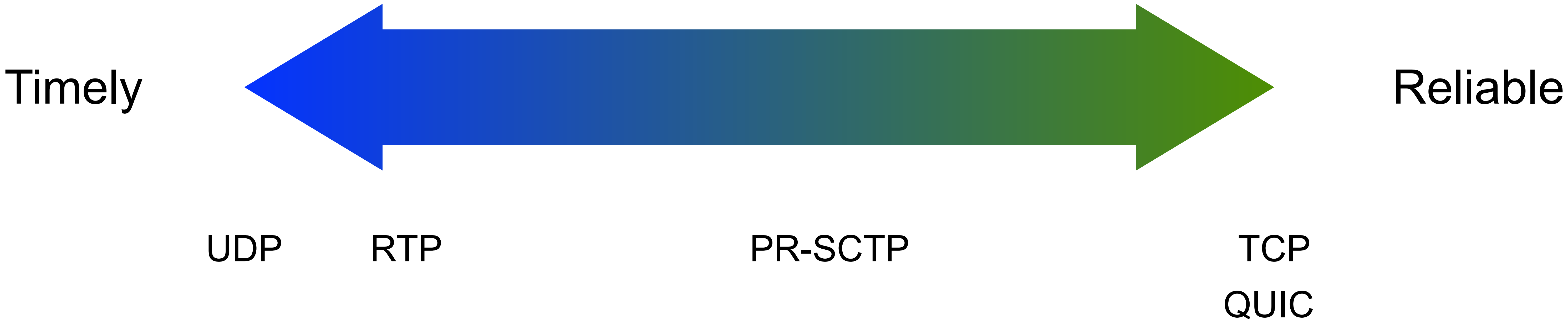
1. INTRODUCTION

Choosing the proper boundaries between functions is perhaps the primary activity of the computer system designer. Design principles that provide guidance in this choice of function placement are among the most important tools of a system designer. This paper discusses one class of function placement argument that has been used for many years with neither explicit recognition nor much conviction. However, the emergence of the data communication network as a computer system component has sharpened this line of function placement argument by making more apparent the situations in which and the reasons why it applies. This paper articulates the argument explicitly, so as to examine its nature and to see how general it really is. The argument appeals to application requirements and provides a rationale for moving a function upward in a layered system closer to the application that uses the function. We begin by considering the communication network version of the argument.

This is a revised version of a paper adapted from End-to-End Arguments in System Design by J. H. Saltzer, D.P. Reed, and D.D. Clark from the 2nd International Conference on Distributed Systems (Paris, France, April 8–10) 1981, pp. 509–512. © IEEE 1981
This research was supported in part by the Advanced Research Projects Agency of the U.S. Department of Defense and monitored by the Office of Naval Research under contract N00014-75-C-0661.
Authors' address: J. H. Saltzer and D. D. Clark, M.I.T. Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 02139. D. P. Reed, Software Arts, Inc., 27 Mica Lane, Wellesley, MA 02181.
Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.
© 1984 ACM 0734-2071/84/1100-0277 $00.75
ACM Transactions on Computer Systems, Vol. 2, No. 4, November 1984, Pages 277–288.

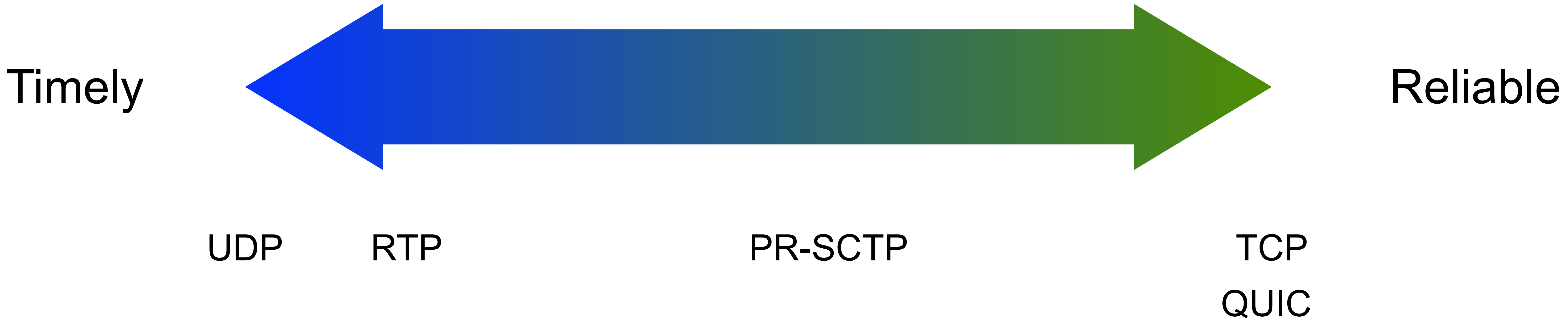
J. Saltzer, D. Reed, and D. Clark, " End-to-end arguments in system design", ACM Transactions on Computer Systems, November 1984. http://dx.doi.org/10.1145/357401.357402

Timeliness vs Reliability Trade-off (1/2)



- Repairing or retransmitting lost packets takes time
- Fundamental trade-off:
  - If a connection is to be reliable, it **cannot** guarantee timeliness
  - If a connection is to be timely, it **cannot** guarantee reliability

Timeliness vs Reliability Trade-off (2/2)



- Different applications make different timeliness vs. reliability trade-offs:
  - Web, email, etc. → data must be delivered in order sent; no strong timeliness requirement
  - Telephony and video conferencing → tolerates some data loss, but requires timeliness
- Implication for network architecture:
  - Network layer should provide a timely but unreliable service
  - Transport layer protocols can add reliability, if needed

Packet Loss in the Internet

- Best effort traffic
- The end-to-end argument
- Timeliness vs. reliability trade-off