Available online at www.sciencedirect.com
SCIENCE DIRECT®
Electronic Notes in Theoretical Computer Science
ELSEVIER Electronic Notes in Theoretical Computer Science 133 (2005) 255–273
www.elsevier.com/locate/entcs

Specifying and Verifying Communications Protocols using Mixed Intuitionistic Linear Logic

David Sinclair[1]

School of Computing
Dublin City University
Glasnevin, Dublin 9, Ireland

James Power[2]

Department of Computer Science
National University of Ireland, Maynooth
Maynooth, Co. Kildare, Ireland

Abstract

In this paper we present a technique for specifying and verifying communications protocols and demonstrate this approach by specifying and verifying two of the fundamental communications protocols, namely TCP and IP, which form the basis of many distributed systems. The logical formalism used is Mixed Intuitionistic Linear Logic in order to use both commutative and non-commutative operators to model the concurrent and sequential processes in these protocols. Key properties of both protocols are proved.

Keywords: complex systems, formal methods, mixed intuitionistic linear logic

1 Introduction

This paper presents an approach for specifying and verifyng communications protocols. This approach will be used to specify and verify the Internet Protocol (IP)[7] and elements of the Transmission Control Protocol (TCP)[8].

[1] Email: David.Sinclair@computing.dcu.ie
[2] Email: James.Power@may.ie

doi:10.1016/j.entcs.2004.08.068

The approach is based on mixed intuitionistic linear logic and describes how this logic can be used to prove some key properties of both protocols. We have previously presented a specification of IP in [10] using commutative linear logic. In this paper we extend this specification considerably to include the specification and verification of TCP. TCP, like many comunications protocols and distributed systems includes both sequential and concurrent processes. Specifying and verifying such systems with the commutative operators of linear logic is difficult. Linear logic is particularly suited to the description of state-based systems since it keeps track of the resources used in each deduction step. Mixed intuitionistic linear logic is a variant of linear logic that contains both commutative and non-commutative operators, and as such is useful where the order of the consumption of resources must be specified. The non-commutative operators of mixed intuitionistic linear logic are ideally suited to specifying systems with both sequential and concurrent processes. The main contribution of this research is to demonstrate how mixed intuitionistic linear logic can be used to specify and verify these types of distributed systems.

In the following sections we briefly describe IP and TCP and mixed intuitionistic linear logic. We then present an outline of our specification of the user interfaces for IP and TCP, demonstrating the role of the linear operators in the axioms. We present a specification of the data transfer component of the TCP protocol; and finally, we outline verification process undertaken to prove key properties of IP and TCP.

1.1 TCP/IP

The Transmission Control Protocol (TCP) and the Internet Protocol (IP) are two essential elements of the communications stack at the heart of many network-based applications. Both of these protocols are typical of state-based distributed systems. IP is responsible for transmitting data from one internet node to another, but does not guarantee the delivery of data to the destination node. TCP is a protocol that sits on top of IP and it has the responsibility of establishing an end-to-end error free connection between peer TCP entities.

IP has no mechanisms to provide end-to-end data reliability, flow control, sequencing, or other services commonly found in host-to-host communications protocols. There are no acknowledgements, either end-to-end or hop-by-hop, and the error detection provided by the IP checksum only covers the IP packet header and not the data itself. In IP there is no flow control or retransmission. IP packets can be lost, duplicated and delivered in any order.

TCP is layered on top of IP and it is its function to establish an error-free end-to-end connection between peer TCP entities. Since IP provides no