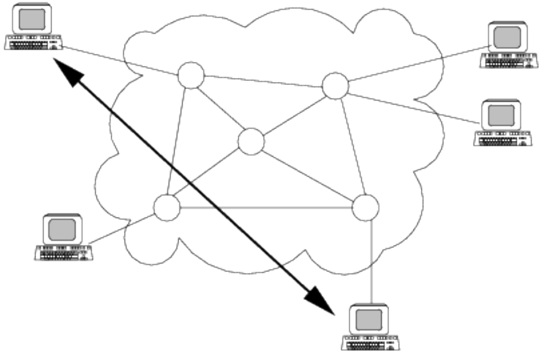
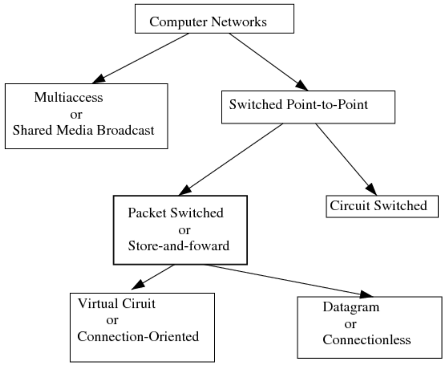
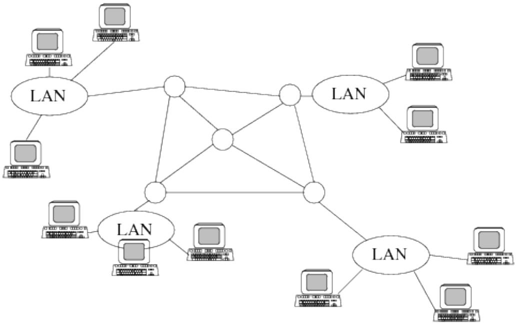
Review



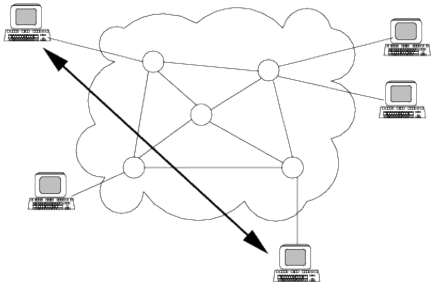
Review


Computer Networks
Multiaccess or Shared Media Broadcast
Switched Point-to-Point
Packet Switched or Store-and-foward
Circuit Switched
Virtual Ciruit or Connection-Oriented
Datagram or Connectionless

Review



How to Send a Message over a Network?



- Routing
- Addressing
- Reliable Data Transfer
- Congestion Control

How to Master Complexity?

- Computer networks are very complex; many issues to address:
  - connection setup, message segmentation, multiplexing, routing, flow control, security, error control, encoding, addressing, ...
- Useful method for dealing with complexity is using "**modularity**".
  - break complex problem into simpler sub-problems
  - use "black box" (input/output) abstraction for sub-problems

How to Master Complexity?

- Computer networks are very complex; many issues to address:
  - connection setup, message segmentation, multiplexing, routing, flow control, security, error control, encoding, addressing, ...
- Useful method for dealing with complexity is using “**modularity**”.
  - break complex problem into simpler sub-problems
  - use “black box” (input/output) abstraction for sub-problems

How to Master Complexity?

- Computer networks are very complex; many issues to address:
    - connection setup, message segmentation, multiplexing, routing, flow control, security, error control, encoding, addressing, ...
- Useful method for dealing with complexity is using “**modularity**”.
    - break complex problem into simpler sub-problems
    - use “black box” (input/output) abstraction for sub-problems

How to Master Complexity?

- Computer networks are very complex; many issues to address:
  - connection setup, message segmentation, multiplexing, routing, flow control, security, error control, encoding, addressing, ...
- Useful method for dealing with complexity is using "**modularity**".
  - break complex problem into simpler sub-problems
  - use "black box" (input/output) abstraction for sub-problems

How to Master Complexity?

- Computer networks are very complex; many issues to address:
  - connection setup, message segmentation, multiplexing, routing, flow control, security, error control, encoding, addressing, ...
- Useful method for dealing with complexity is using “**modularity**”.
  - break complex problem into simpler sub-problems
  - use “black box” (input/output) abstraction for sub-problems

SMTP (Simple Mail Transfer Protocol)

S: 220 sf.com

C: HELO toronto.edu

S: 250 Hello toronto.edu, pleased to meet you

C: MAIL FROM: <alice@toronto.edu>

S: 250 alice@toronto.edu... Sender ok

C: RCPT TO: <bob@sf.com>

S: 250 bob@sf.com ... Recipient ok

C: DATA

S: 354 Enter mail, end with "." on a line by itself

C: How are you?

C: See you soon.

C: .

S: 250 Message accepted for delivery

C: QUIT

S: 221 sf.com closing connection

Modularity for Computer Networks

- **Hierarchical Layering:** The type of functional modularity used for computer networks is hierarchical layering. What is special about this architecture is that it is distributed and connected through unreliable links with delays.
- Example: Postal Service
  - When I bring a letter to the post office, I don't know how it gets delivered from there. The office clerk doesn't know the exact details either, and so on.

Modularity for Computer Networks

- **Hierarchical Layering:** The type of functional modularity used for computer networks is hierarchical layering. What is special about this architecture is that it is distributed and connected through unreliable links with delays.
- Example: Postal Service
  - When I bring a letter to the post office, I don't know how it gets delivered from there. The office clerk doesn't know the exact details either, and so on.

Modularity for Computer Networks

- **Hierarchical Layering:** The type of functional modularity used for computer networks is hierarchical layering. What is special about this architecture is that it is distributed and connected through unreliable links with delays.
- Example: Postal Service
    - When I bring a letter to the post office, I don't know how it gets delivered from there. The office clerk doesn't know the exact details either, and so on.

Layered Network Architecture

Application Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer

Layered Network Architecture

Application Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer

There are several ways to define a layered network architecture. In this course, we consider the 5 Internet layers. Another model consists of the 7 OSI layers.

Outline

- Description of the different network layers
- Issues in layered network architecture

$->$ Read Chapter 1 in Textbook

Goals

- Know what the different layers do
- Know how layers interact
- Terminology: peer process, protocol, service

Application Layer

- **Service:** Supports applications
- **Tasks:**
  - Connection Setup
  - Flow control
  - Error control
- **Protocols:** HTTP to support Web, SMTP to support email, FTP to support file transfer.
- **Location**: End Systems/Hosts

Application Layer

- **Service:** Supports applications
- **Tasks:**
  - Connection Setup
  - Flow control
  - Error control
- **Protocols:** HTTP to support Web, SMTP to support email, FTP to support file transfer.
- **Location**: End Systems/Hosts

Application Layer

- **Service:** Supports applications
- **Tasks:**
  - Connection Setup
  - Flow control
  - Error control
- **Protocols:** HTTP to support Web, SMTP to support email, FTP to support file transfer.
- **Location:** End Systems/Hosts

Application Layer

- **Service:** Supports applications
- **Tasks:**
  - Connection Setup
  - Flow control
  - Error control
- **Protocols:** HTTP to support Web, SMTP to support email, FTP to support file transfer.
- **Location**: End Systems/Hosts

SMTP (Simple Mail Transfer Protocol)

S: 220 sf.com

C: HELO toronto.edu

S: 250 Hello toronto.edu, pleased to meet you

C: MAIL FROM: <alice@toronto.edu>

S: 250 alice@toronto.edu... Sender ok

C: RCPT TO: <bob@sf.com>

S: 250 bob@sf.com ... Recipient ok

C: DATA

S: 354 Enter mail, end with "." on a line by itself

C: How are you?

C: See you soon.

C: .

S: 250 Message accepted for delivery

C: QUIT

S: 221 sf.com closing connection

Transport Layer

- **Service:** Prepares messages for being transported over the network.
- **Tasks:**
  - Message fragmentation and reassembly
  - Flow Control
  - Congestion control
  - Error control
  - Connection setup
- **Protocols:** TCP (Transmission Control Protocol), UDP (User Datagram Protocol)
- **Location**: End Systems/Hosts

Transport Layer

- **Service:** Prepares messages for being transported over the network.
- **Tasks:**
  - Message fragmentation and reassembly
  - Flow Control
  - Congestion control
  - Error control
  - Connection setup
- **Protocols:** TCP (Transmission Control Protocol), UDP (User Datagram Protocol)
- **Location**: End Systems/Hosts

Transport Layer

- **Service:** Prepares messages for being transported over the network.
- **Tasks:**
  - Message fragmentation and reassembly
  - Flow Control
  - Congestion control
  - Error control
  - Connection setup
- **Protocols:** TCP (Transmission Control Protocol), UDP (User Datagram Protocol)
- **Location:** End Systems/Hosts

Transport Layer

- **Service:** Prepares messages for being transported over the network.
- **Tasks:**
  - Message fragmentation and reassembly
  - Flow Control
  - Congestion control
  - Error control
  - Connection setup
- **Protocols:** TCP (Transmission Control Protocol), UDP (User Datagram Protocol)
- **Location**: End Systems/Hosts

Network Layer

- **Service:** Sends data units over the network
- **Tasks:**
  - Routing
  - Addressing
  - Congestion control
- **Protocols:** IP (Internet Protocol)
- **Location**: End Systems/Hosts + Routers

Network Layer

- **Service:** Sends data units over the network
- **Tasks:**
  - Routing
  - Addressing
  - Congestion control
- **Protocols:** IP (Internet Protocol)
- **Location:** End Systems/Hosts + Routers

Network Layer

- **Service:** Sends data units over the network
- **Tasks:**
  - Routing
  - Addressing
  - Congestion control
- **Protocols:** IP (Internet Protocol)
- **Location:** End Systems/Hosts + Routers

Network Layer

- **Service:** Sends data units over the network
- **Tasks:**
  - Routing
  - Addressing
  - Congestion control
- **Protocols:** IP (Internet Protocol)
- **Location**: End Systems/Hosts + Routers

Data Link Layer

- **Service:** Sends data units over a link
- **Tasks:**
  - Framing
  - Error control
  - Retransmissions
- **Protocols:** ARQ (Automatic Repeat Request), CSMA/CD for Ethernet and Wave LAN.
- **Location**: End Systems/Hosts + Routers

Data Link Layer

- **Service:** Sends data units over a link
- **Tasks:**
  - Framing
  - Error control
  - Retransmissions
- **Protocols:** ARQ (Automatic Repeat Request), CSMA/CD for Ethernet and Wave LAN.
- **Location**: End Systems/Hosts + Routers

Data Link Layer

- **Service:** Sends data units over a link
- **Tasks:**
  - Framing
  - Error control
  - Retransmissions
- **Protocols:** ARQ (Automatic Repeat Request), CSMA/CD for Ethernet and Wave LAN.
- **Location:** End Systems/Hosts + Routers

Data Link Layer

- **Service:** Sends data units over a link
- **Tasks:**
  - Framing
  - Error control
  - Retransmissions
- **Protocols:** ARQ (Automatic Repeat Request), CSMA/CD for Ethernet and Wave LAN.
- **Location**: End Systems/Hosts + Routers

Physical Layer

- **Service:** Sends bits over a link
- **Tasks:**
    - Modem (Modulator/Demodulator)
- **Location**: End Systems/Hosts + Routers

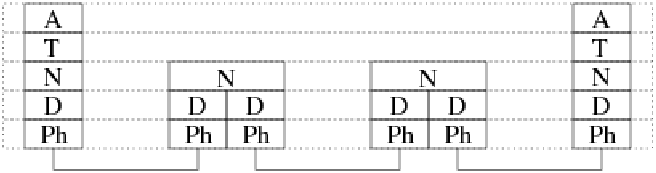
Physical Layer

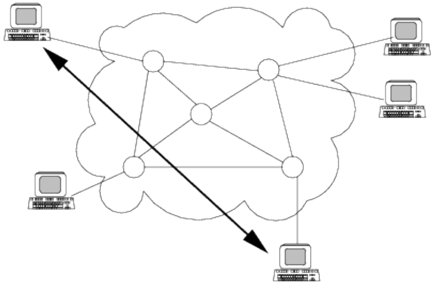
- **Service:** Sends bits over a link
- **Tasks:**
  - Modem (Modulator/Demodulator)
- **Location:** End Systems/Hosts + Routers

Physical Layer

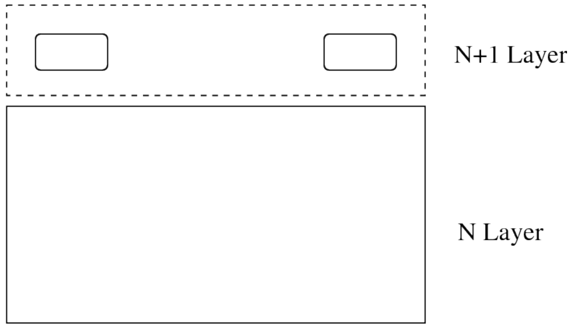
- **Service:** Sends bits over a link
- **Tasks:**
  - Modem (Modulator/Demodulator)
- **Location**: End Systems/Hosts + Routers

“The Roller-Coaster Ride of a Message”





A Closer Look



Terminology

- **Peer or Peer Processes:**
  Members of the same layer at different locations
- **Protocol:**
  Set of rules for how peers interact
- **Protocol Stack:**
  Set of protocol used (one per layer)
- **Network Architecture or Network Reference Model:**
  Set of layers that used for a network.

Terminology

- **Peer or Peer Processes:**
  Members of the same layer at different locations
- **Protocol:**
  Set of rules for how peers interact
- **Protocol Stack:**
  Set of protocol used (one per layer)
- **Network Architecture or Network Reference Model:**
  Set of layers that used for a network.

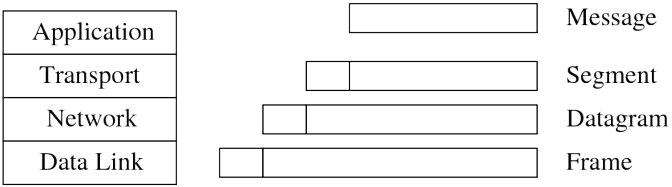
Terminology

- **Peer or Peer Processes:**
  Members of the same layer at different locations
- **Protocol:**
  Set of rules for how peers interact
- **Protocol Stack:**
  Set of protocol used (one per layer)
- **Network Architecture or Network Reference Model:**
  Set of layers that used for a network.

Terminology

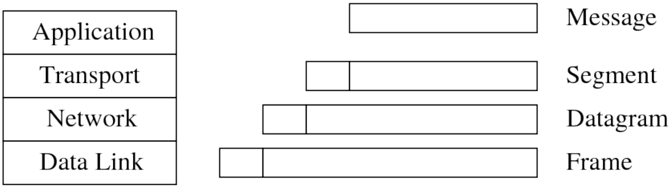
- **Peer or Peer Processes:**
  Members of the same layer at different locations
- **Protocol:**
  Set of rules for how peers interact
- **Protocol Stack:**
  Set of protocol used (one per layer)
- **Network Architecture or Network Reference Model:**
  Set of layers that used for a network.

“Growing” Data Units



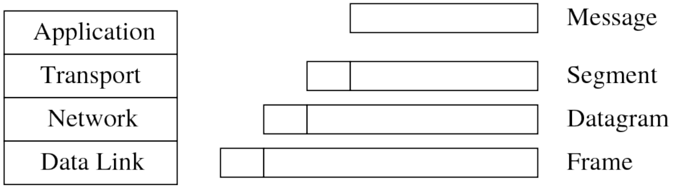
- Protocol Data Unit (PDU)
- Protocol Control Information (PCI) or Header
- Service Data Unit (SDU)

“Growing” Data Units



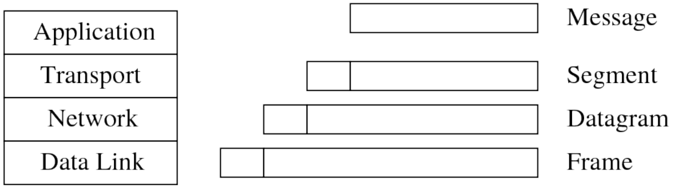
- Protocol Data Unit (PDU)
- Protocol Control Information (PCI) or Header
- Service Data Unit (SDU)

“Growing” Data Units



- Protocol Data Unit (PDU)
- Protocol Control Information (PCI) or Header
- Service Data Unit (SDU)

“Growing” Data Units



- Protocol Data Unit (PDU)
- Protocol Control Information (PCI) or Header
- Service Data Unit (SDU)

Question: Why Layers?

Layer Functions

- **Error Control:** makes the logical channel between two peer processes reliable.
- **Flow Control:** avoids overwhelming a slower peer process with protocol data units.
- **Segmentation and Reassembly of Data Units**
- **Multiplexing:** allows several higher-level sessions to share a single lower-level connection.
- **Connection Setup:** provides handshaking between peer processes.

Layer Functions

- **Error Control:** makes the logical channel between two peer processes reliable.
- **Flow Control:** avoids overwhelming a slower peer process with protocol data units.
- **Segmentation and Reassembly of Data Units**
- **Multiplexing:** allows several higher-level sessions to share a single lower-level connection.
- **Connection Setup:** provides handshaking between peer processes.

Layer Functions

- **Error Control:** makes the logical channel between two peer processes reliable.
- **Flow Control:** avoids overwhelming a slower peer process with protocol data units.
- **Segmentation and Reassembly of Data Units**
- **Multiplexing:** allows several higher-level sessions to share a single lower-level connection.
- **Connection Setup:** provides handshaking between peer processes.

Layer Functions

- **Error Control:** makes the logical channel between two peer processes reliable.
- **Flow Control:** avoids overwhelming a slower peer process with protocol data units.
- **Segmentation and Reassembly of Data Units**
- **Multiplexing:** allows several higher-level sessions to share a single lower-level connection.
- **Connection Setup:** provides handshaking between peer processes.

Layer Functions

- **Error Control:** makes the logical channel between two peer processes reliable.
- **Flow Control:** avoids overwhelming a slower peer process with protocol data units.
- **Segmentation and Reassembly of Data Units**
- **Multiplexing:** allows several higher-level sessions to share a single lower-level connection.
- **Connection Setup:** provides handshaking between peer processes.

Service Types

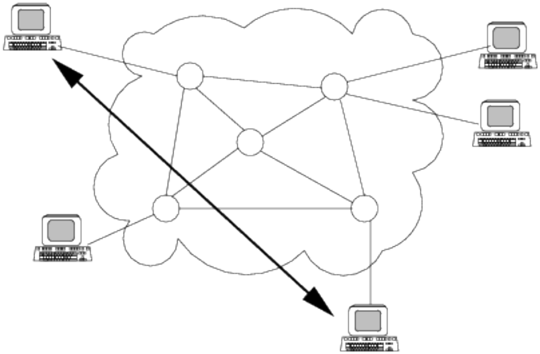
- **Connection-Oriented:** Connection setup through handshaking. After connection setup, data messages can be exchanged. During the handshaking, parameters used in the protocol can be exchanged/negotiated (to provide reliable data transfer, flow control, congestion control, etc.).
- **Connectionless:** No connection setup. Data messages are sent immediately. A connectionless service is by its nature unreliable.

Service Types

- **Connection-Oriented:** Connection setup through handshaking. After connection setup, data messages can be exchanged. During the handshaking, parameters used in the protocol can be exchanged/negotiated (to provide reliable data transfer, flow control, congestion control, etc.).
- **Connectionless:** No connection setup. Data messages are sent immediately. A connectionless service is by its nature unreliable.

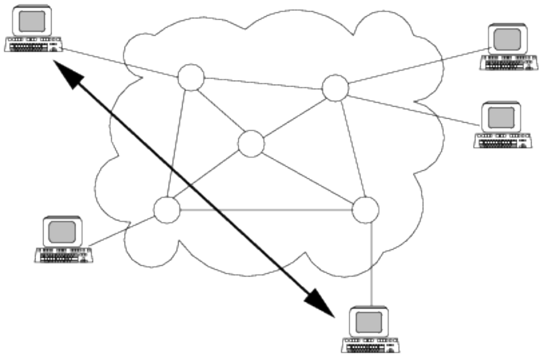
Distributed Protocols

Issues



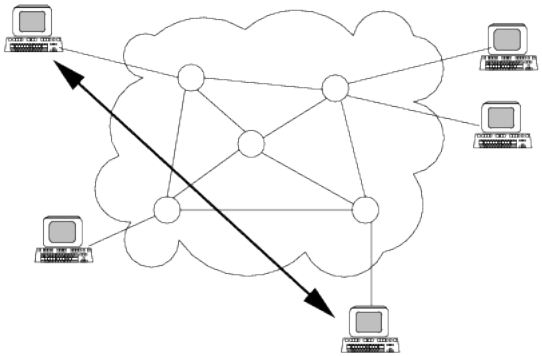
- Reliability
- Performance/Quality-of-Service (QoS)

Issues



- Reliability
- Performance/Quality-of-Service (QoS)

Issues



- Reliability
- Performance/Quality-of-Service (QoS)

Quality of Service

Factors Determining the Quality of Service

- Delay
- Packet Loss
- Transmission Rate

Quality of Service

Factors Determining the Quality of Service

- Delay
- Packet Loss
- Transmission Rate

Quality of Service

Factors Determining the Quality of Service

- Delay
- Packet Loss
- Transmission Rate

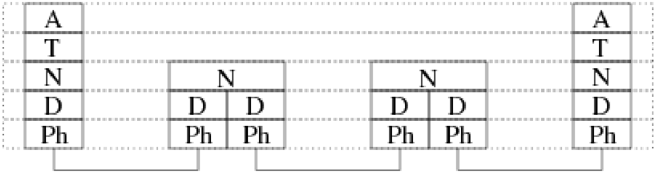
Quality of Service

Factors Determining the Quality of Service

- Delay
- Packet Loss
- Transmission Rate

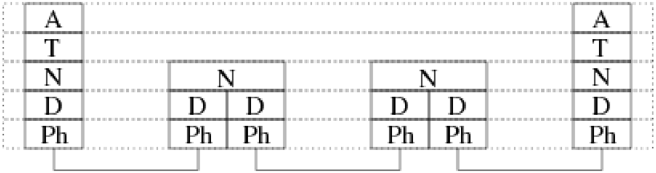
Why is Quality of Service Important?

Delay



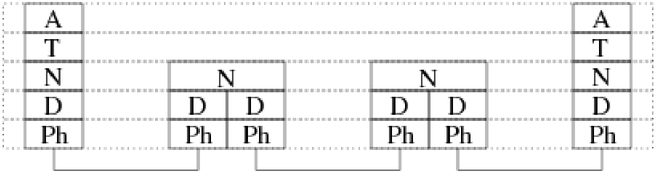
- Processing Delay
- Queueing Delay
- Transmission Delay
- Propagation Delay

Delay



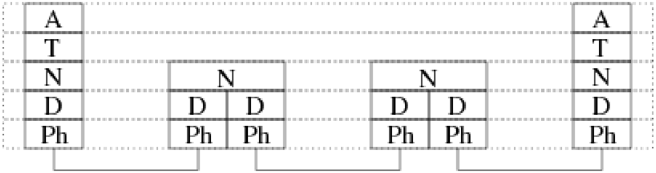
- Processing Delay
- Queueing Delay
- Transmission Delay
- Propagation Delay

Delay



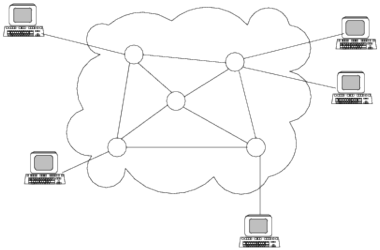
- Processing Delay
- Queueing Delay
- Transmission Delay
- Propagation Delay

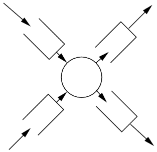
Delay



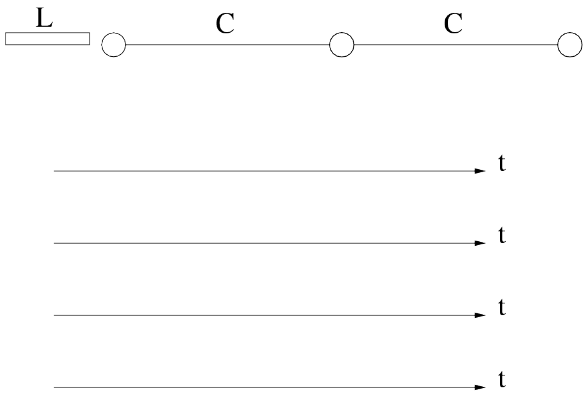
- Processing Delay
- Queueing Delay
- Transmission Delay
- Propagation Delay

Queueing Delay





Transmission Delay vs. Propagation Delay



Approach

1. Reliable Data Transfer
2. Tools for Performance Analysis/QoS Evaluation
3. Modelling and Analysis of Protocols
4. Implementation Issues