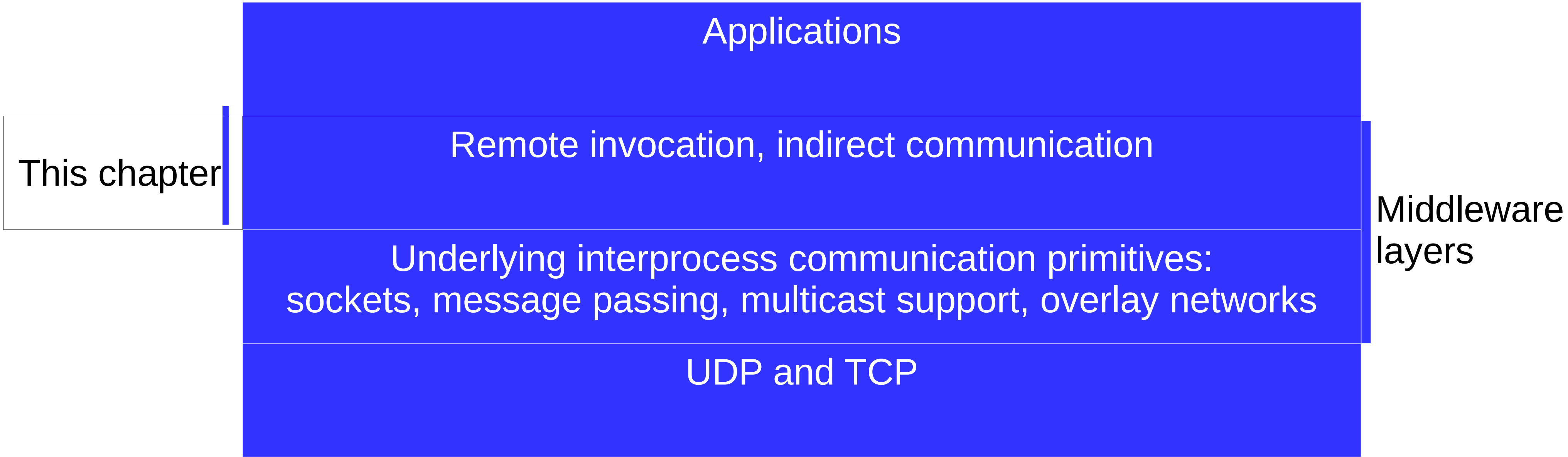
03 – Remote invocaction

- Request-reply
- RPC
- RMI

- Coulouris 5
- Birrel_Nelson_84.pdf

Remote invocation

- Mechanisms for process communication on a DS
- Built on top of interprocess communication primitives
  - Lower level
  - Provided by the OS

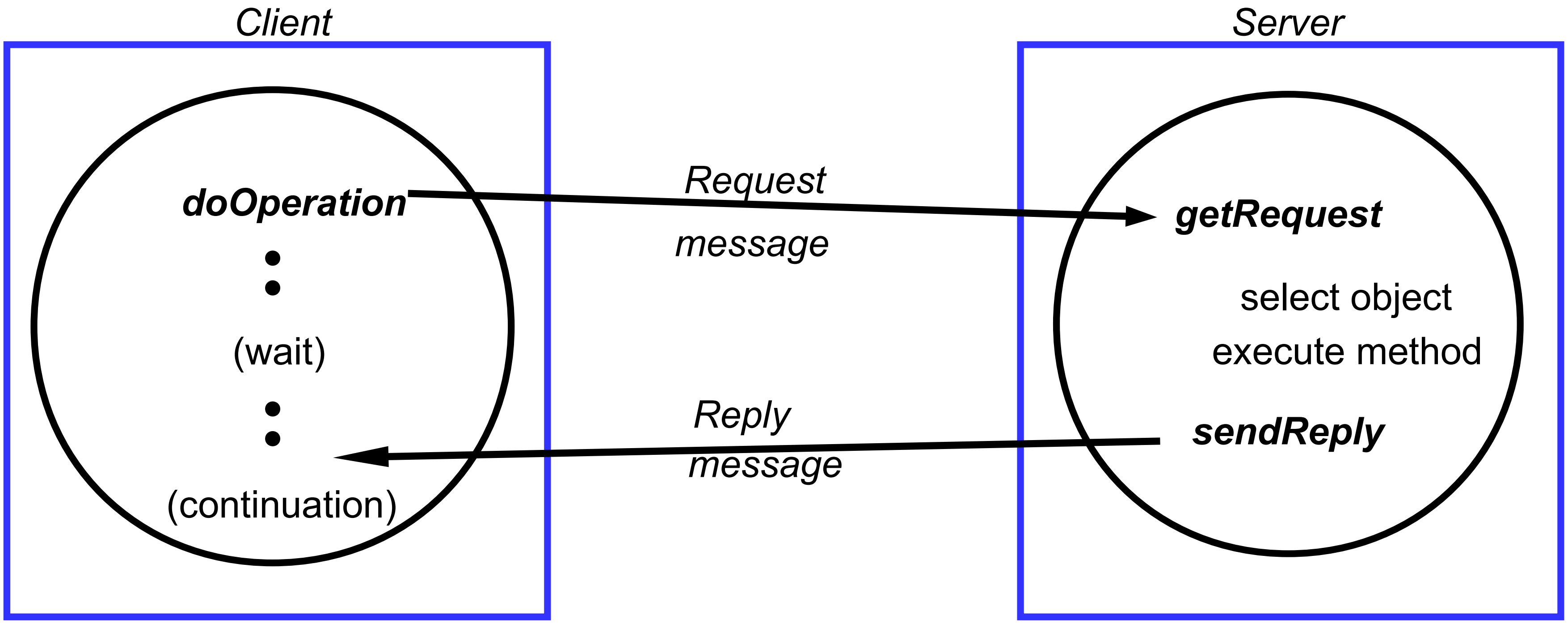

Applications
This chapter
Remote invocation, indirect communication
Underlying interprocess communication primitives:
sockets, message passing, multicast support, overlay networks
Middleware layers
UDP and TCP

Request Reply Protocols

- Support client-server interaction
- Usually synchronous
  - Client blocks until reply is received
- Reliable (reply is an acknowledge)
- May use UDP datagram instead of TCP streams
  - TCP knowledge is redundant
  - Connection establishing requires extra messages
  - Flow control is redundant for some uses
    - Which pass only small arguments and results

Request Reply Protocols

- Communication primitives
  - doOperation
  - getRequest
  - sendReply



Request Reply Protocols

- doOperation
  - Sends request to the remote server
  - Arguments
    - Server remote reference
    - Operation id
    - Arguments
  - Returns
    - reply
  - send (TCP), sendto (UDP)

Request Reply Protocols

- getRequest
  - Blocking function / waits for request
  - Returns request data
- sendReply
  - Sends response to client
  - Arguments
    - Reply
    - Client id
  - send (TCP), sendto (UDP)

Request Reply Protocols

MessageType	int (0=Request, 1= Reply)
requestId	int
remoteReference	RemoteRef
operationID	int or Operation
arguments	array of bytes

- Message identifier
  - RequestId
    - increasing sequence of integers (per sending process)
  - RemoteReference
    - Identification of the process (IP address, port, process id)

Request Reply Protocols

- Failure model (UDP)
  - Omission failures
  - Delivery order not guaranteed
- Timeouts
  - doOperation can
    - Repeats transmission of message

Request Reply Protocols

- Duplicate messages
  - After time-outs
  - Protocols must recognize them
-
  - Reception of duplicate request
  - Ignore second repetitions
  - Implement idempotent operation
- History
  - required

Request Reply Protocols

- Exchange protocols
  - R (request)
    - Online request is sent
    - No need for response
    - Client may unblock imediatly
  - RR (request-reply)
    - Special ACK message not required
      - Implemented by the response
    - Subsequent Request is Acknowledgment of the reply

- Exchange protocols
  - RRA (request-reply-ack)
    - Explicit ack of the reply
    - Server can clean history
- TCP
  - simplifies protocol
  - Larger messages
  - Acknowledgements not necessary