

# MYE017 Distributed Systems

Kostas Magoutis

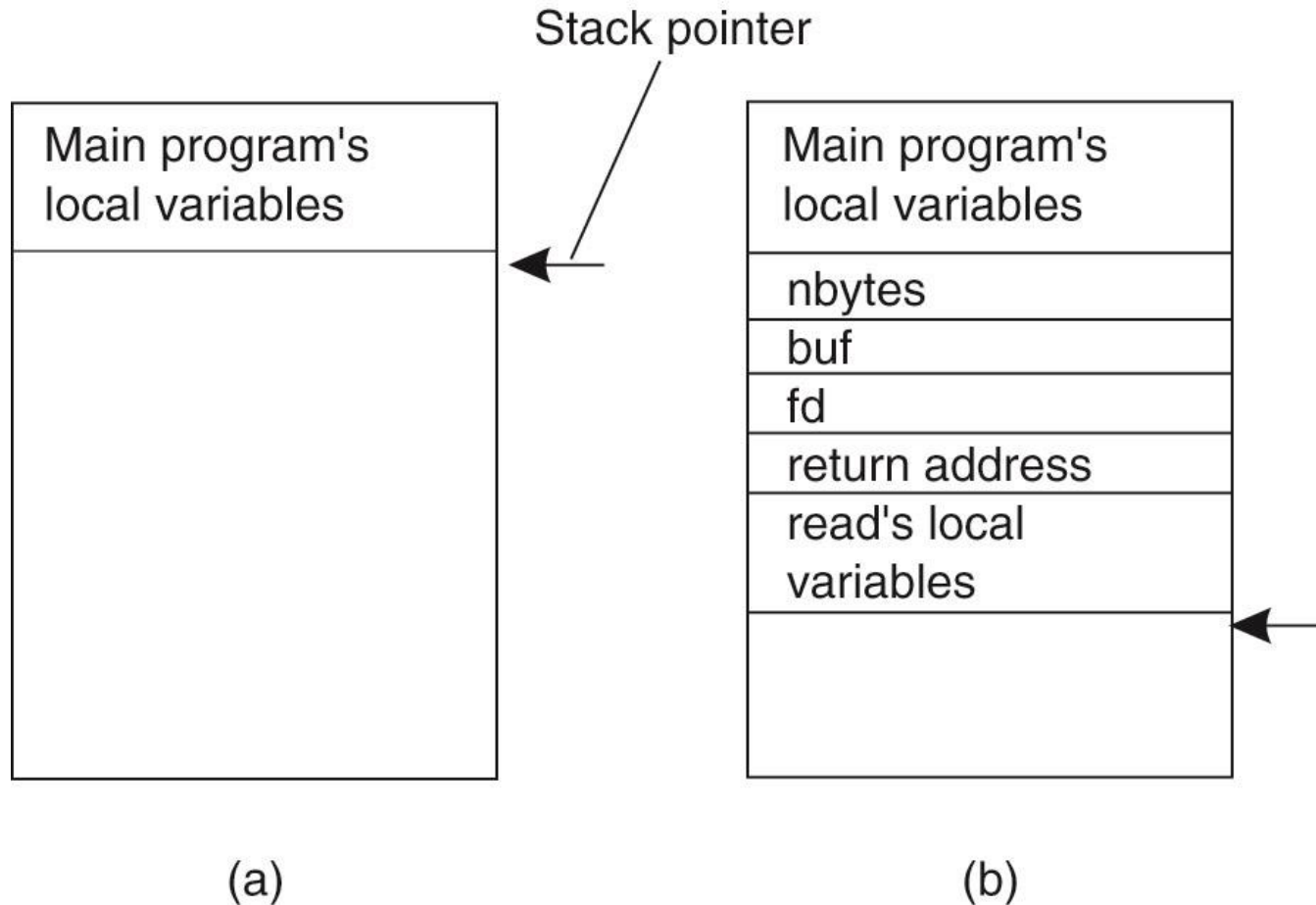
magoutis@cse.uoi.gr

<http://www.cse.uoi.gr/~magoutis>

# Transmission control protocol (TCP)

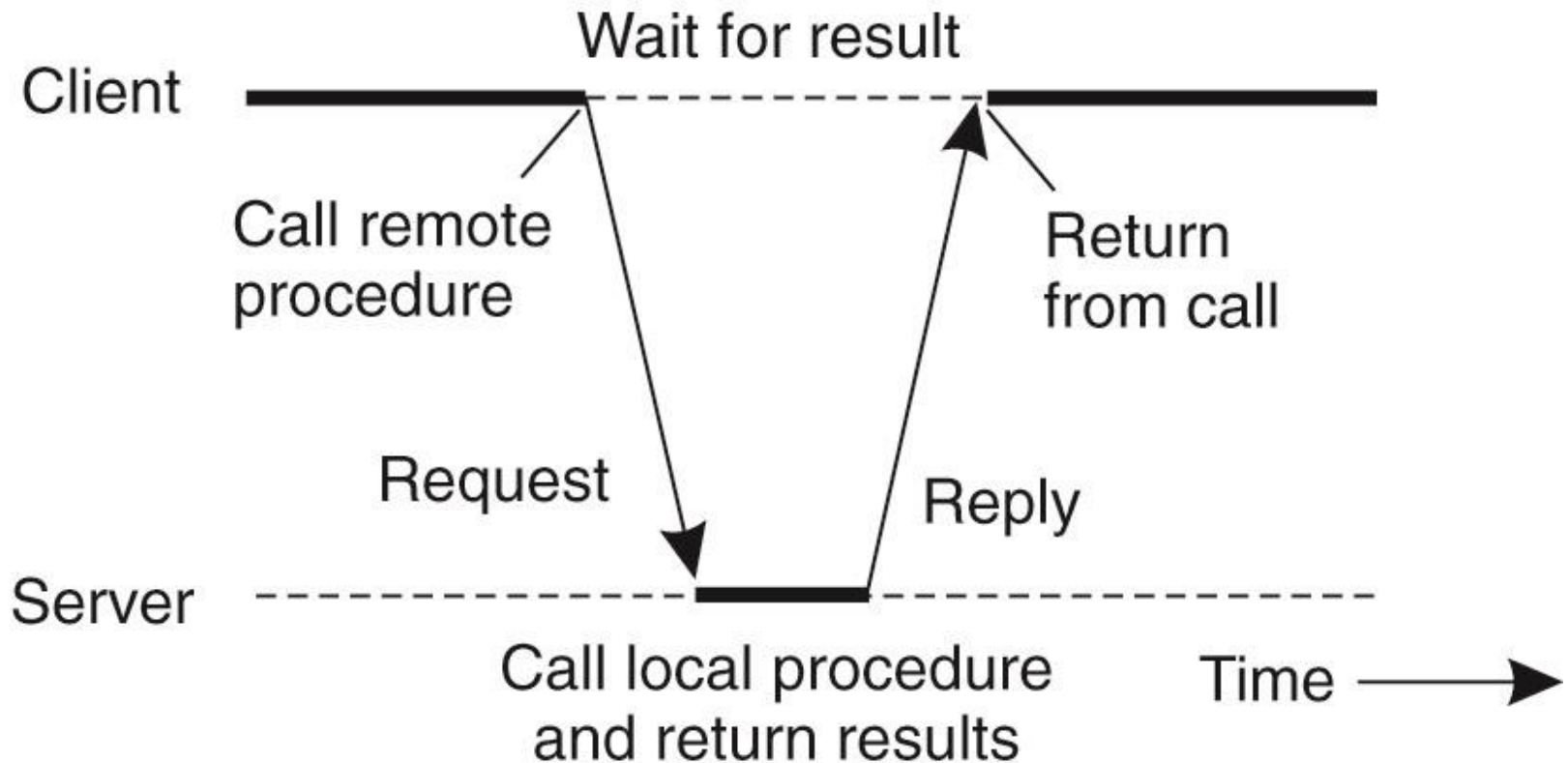
- Connection oriented
- Point-to-point
- Byte stream
- Reliable
- In-order

# Conventional Procedure Call



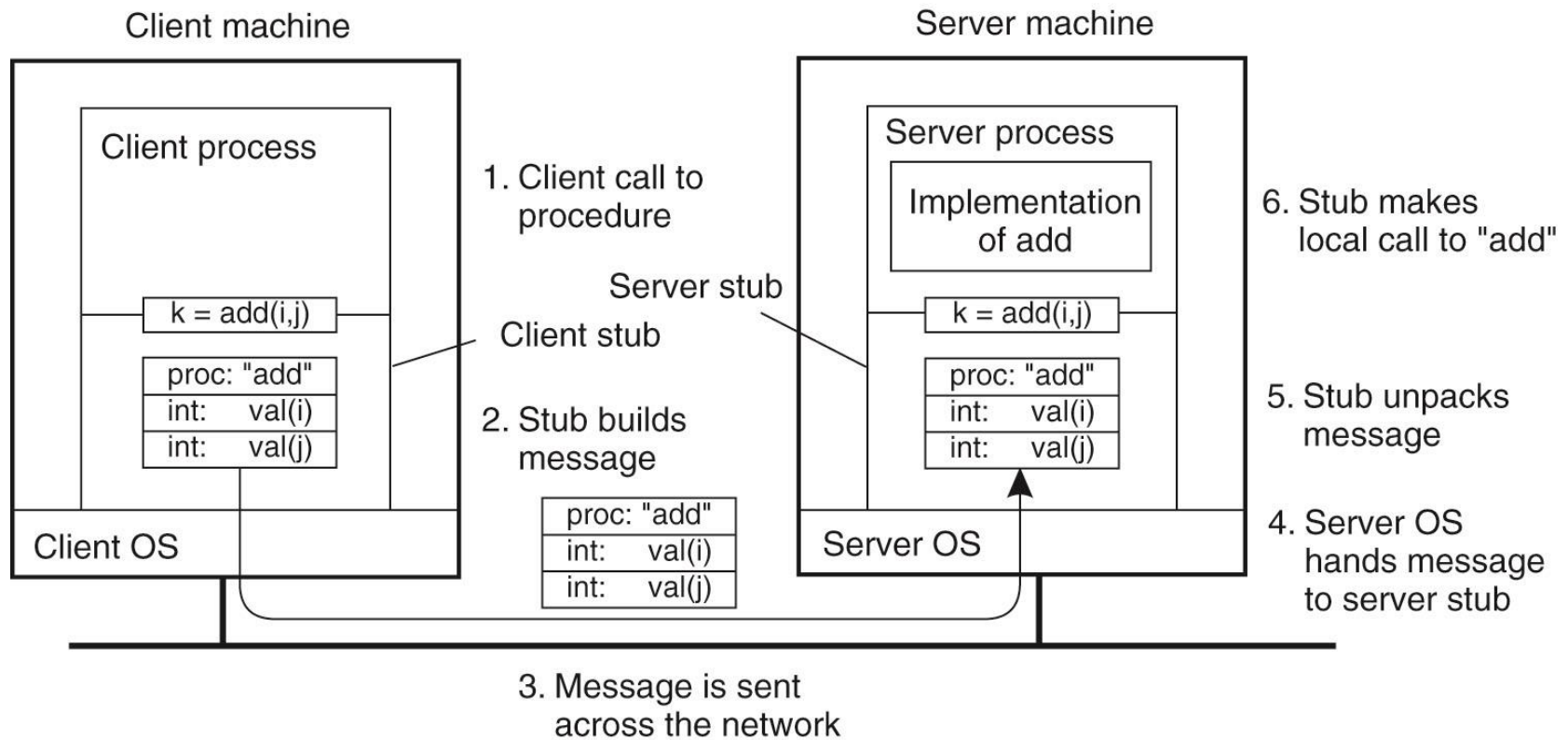
(a) Parameter passing in a local procedure call: the stack before the call to read. (b) The stack while the called procedure is active.

# Client and Server Stubs



Principle of RPC between a client and server program.

# Passing Value Parameters (1)



The steps involved in a doing a remote computation through RPC.

# Parameter Specification and Stub Generation

```
foobar( char x; float y; int z[5] )  
{  
  ....  
}
```

(a)

foobar's local variables	
	x
y	
5	
z[0]	
z[1]	
z[2]	
z[3]	
z[4]	

(b)

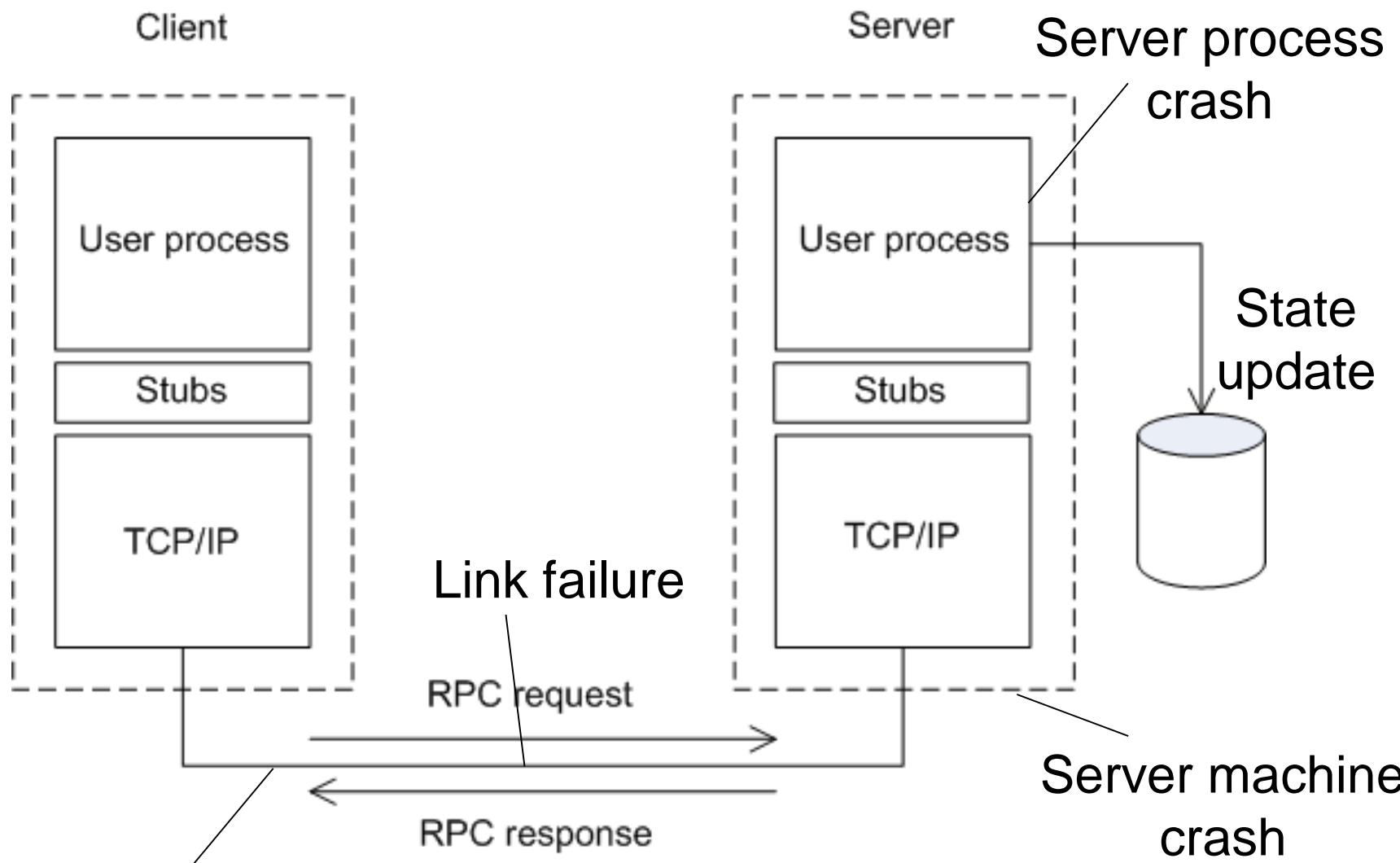
(a) A procedure. (b) The corresponding message.

# Failure Models

Type of failure	Description
Crash failure	A server halts, but is working correctly until it halts
Omission failure <i>Receive omission</i> <i>Send omission</i>	A server fails to respond to incoming requests A server fails to receive incoming messages A server fails to send messages
Timing failure	A server's response lies outside the specified time interval
Response failure <i>Value failure</i> <i>State transition failure</i>	The server's response is incorrect The value of the response is wrong The server deviates from the correct flow of control
Arbitrary failure	A server may produce arbitrary responses at arbitrary times

Different types of failures.

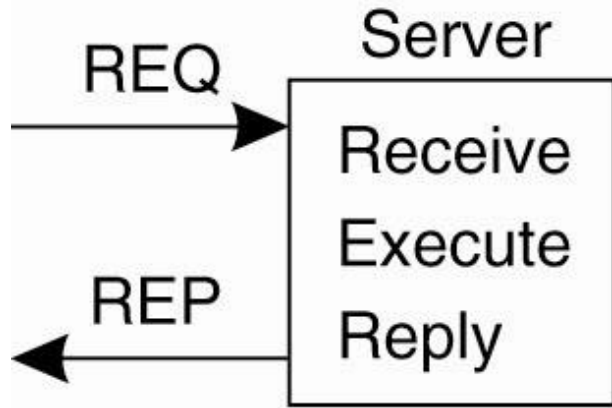
# RPC behavior under failures



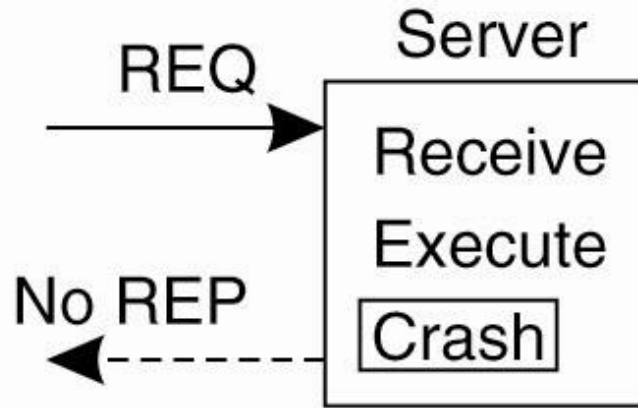
Message / packet omission failures handled by TCP



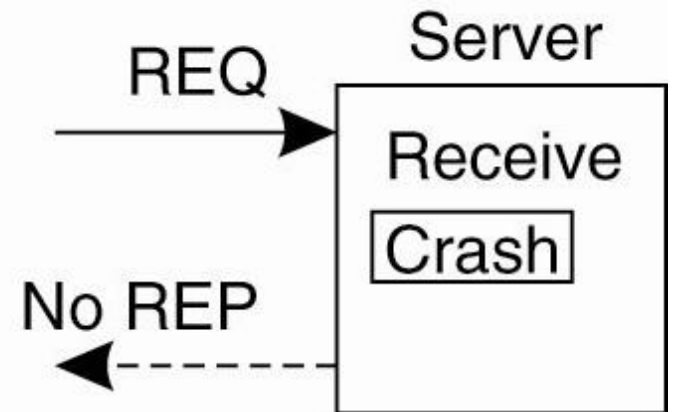
# Server Crashes



(a)



(b)



(c)

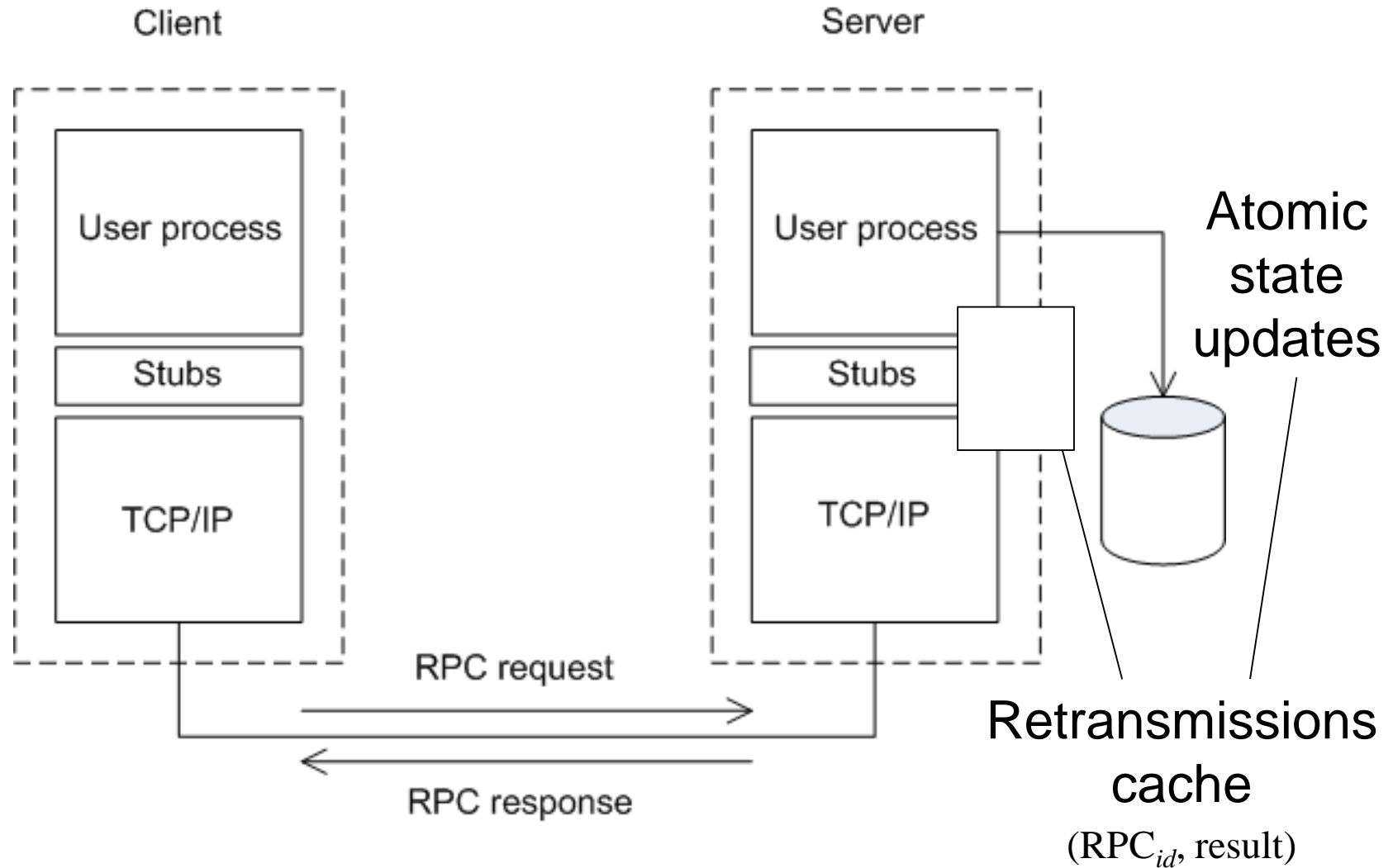
A server in client-server communication

- (a) The normal case.
- (b) Crash after execution.
- (c) Crash before execution.

# RPC semantics

- *At least once*
  - Retry after an exception until successful
  - Good choice with idempotent operations
- *At most once*
  - Do not retry after an exception
- *Exactly once*
  - Return error if not possible

# Towards *exactly once*



# Client crashes after sending request

- Problem: Creates *orphan* server-side work
  - Wastes CPU, ties-up locks or other resources
- Possible solutions
  - Log request, clean up during client recovery
  - Client epochs: Servers discard orphans after receiving *new-epoch* broadcast
  - Discard only if owner cannot be located
  - Give each RPC time  $T$  to do the job; if not done by that time, ask for extension  $T$  or die