Network services

Written by [Febogamedev]

Network services are characterized by two attributes:

- Whether a connection exists between the sender and the receiver.
- The reliability of the service.

The first attribute indicates whether the network service is connection-oriented or connectionless (Fairhurst et al., 2017).
The second attribute indicates whether the service is reliable or unreliable (Fairhurst et al., 2017).

Based on these attributes, a service can be (Fairhurst et al., 2017):

- A connection-oriented and unreliable service.
- A connectionless and unreliable service.
- A connection-oriented and reliable service.
- A connectionless and reliable service.

A Connection-Oriented and Unreliable Service
An example of a connection-oriented but unreliable service is real-time audio or video communication, such as a video call (Schulzrinne et al., 2003).
The service is connection-oriented because a session is established between the two devices before data is exchanged: the call is initiated, accepted, and kept active until it ends. However, the service can be considered unreliable because it does not necessarily guarantee the retransmission of every lost audio or video segment (Eggert et al., 2017; Schulzrinne et al., 2003).
In real-time communication, retransmitting data that arrives too late would be of little use. If a small audio packet is lost, the system may prefer to ignore or compensate for it rather than interrupt the communication to recover it (Eggert et al., 2017).

A Connectionless and Unreliable Service
An example of a connectionless and unreliable service is the transmission of a real-time location in a mapping, delivery, or tracking app (Eggert et al., 2017).
The device periodically sends small location updates, for example: “I am at this location,” followed a few seconds later by “I am now at this new location.” Each update is independent of the others and is mainly useful at the moment it is sent.
If one of these updates is lost, the system may not retransmit it because newer information will arrive shortly afterward. In this case, receiving the updated location quickly is more useful than recovering an outdated one (Eggert et al., 2017).

A Connection-Oriented and Reliable Service
An example of a connection-oriented and reliable service is uploading a file to a cloud service such as Google Drive or Dropbox (Eddy, 2022).
When we upload a file, our device establishes a logical connection with the service’s server (Eddy, 2022).

During the transfer, data is sent in a controlled manner so that the server can receive it correctly (Eddy, 2022).
This service is reliable because the file must arrive complete and intact.
If part of the data is lost or arrives incorrectly, the system must detect the problem and retransmit the missing information. Otherwise, the file may be corrupted or impossible to open (Eddy, 2022).

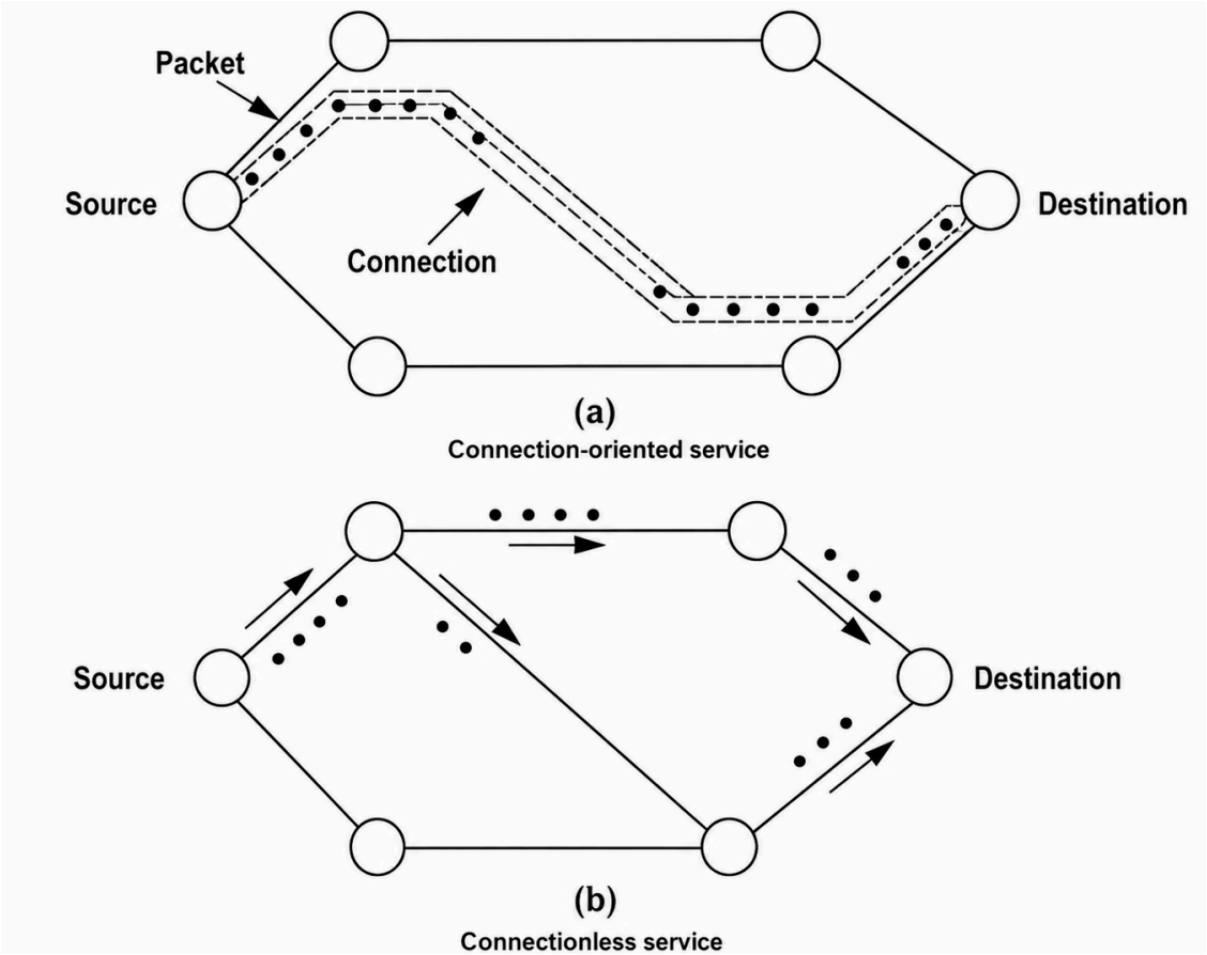
A Connectionless and Reliable Service
An example of a connectionless and reliable service is an app sending an important notification, such as a banking or delivery notification (Thomson et al., 2016).
The message can be sent as an independent unit: continuous communication between sender and receiver does not need to be maintained. Each notification contains the information required for delivery to the correct device.
However, the service can be considered reliable because the system attempts to ensure that the notification is actually delivered. It may use delivery acknowledgments, retransmissions, or temporary storage: if the device is unreachable at that moment, the notification can be stored and sent as soon as the device comes back online Thomson et al., 2016).

Connection

In some respects, **connection-oriented** services resemble circuit switching: before data is exchanged, a logical connection is established between the sender and the receiver (Fairhurst et al., 2017):

- A connection is established, creating a “communication channel” between the source and the destination through appropriate means. This process typically involves several devices along the path from the source to the destination.
- Once established, the connection acts like a digital pipe through which all transmitted data flows, arriving in the same order in which it was sent (Eddy, 2022).
- When communication ends, the connection is released, once again involving all the devices along the path.

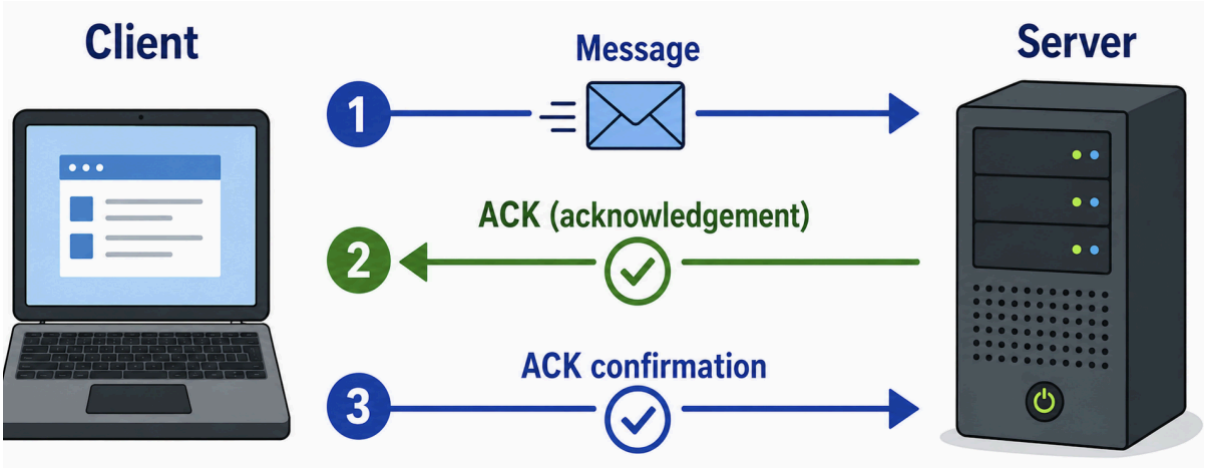
Connectionless services, by contrast, are modeled on the postal system: each letter travels independently of the others; it arrives when it arrives, and it may not arrive at all. Furthermore, two letters with the same sender and receiver may travel along different routes (Fairhurst et al., 2017; Postel, 1980).
In the image below, a connection-oriented service is shown at the top and a connectionless service at the bottom.



A connection-oriented service at the top and a connectionless service at the bottom

Reliability

A service is generally characterized as either reliable or unreliable.
A reliable service attempts to ensure that data arrives complete, correct, and in the expected order. To achieve this, it uses acknowledgments, error checking, and retransmissions when necessary (Eddy, 2022; Fairhurst et al., 2017).
Generally, a reliable service requires the receiver to send an acknowledgment to the source for every packet received (Eddy, 2022).



This image shows an example of a reliable service: whenever a client sends a message, it must receive an acknowledgment from the server (ACK), which in turn must receive confirmation that the previous acknowledgment was received (Eddy, 2022).

This reliability introduces **overhead** because ensuring reliability requires additional computational resources. In some situations, this may be undesirable, for example in a real-time application such as video streaming (Eggert et al., 2017; Schulzrinne et al., 2003).

Conversely, an **unreliable service** does not guarantee that specific data will actually reach its destination (Fairhurst et al., 2017; Postel, 1980).

Supplementary materials

Exercises

Which Service Should Be Chosen?

A company is designing four functions for a new application:

- A video call between two users.
- The periodic transmission of a courier's location.
- The upload of an important document to the cloud.
- The delivery of a banking notification that must reach the user even if the phone is offline at that moment.

For each function:

- Determine whether the service should be connection-oriented or connectionless.
- Determine whether it should be reliable or unreliable.
- Classify each case into one of the four combinations covered in the learning material.
- Justify each choice by explaining what would happen if some data were lost.
- Indicate in which cases retransmission would be useful and in which cases it might instead be inefficient.
- Explain why no combination is always better than the others, and why the choice depends on the service being designed.

A Video Call with Network Problems

During a video call, some small audio fragments are lost.
The system designer must choose between two strategies:

- Strategy A: temporarily pause the communication and retransmit every missing fragment.
- Strategy B: continue the communication immediately even if some fragments are lost.

Analyze the two solutions.

- Indicate which strategy is more consistent with the type of real-time service described in the learning material.
- Explain why a video call can be considered connection-oriented.
- Explain why it can still use an unreliable service for part of the data.
- Explain why receiving an audio fragment with a long delay may be less useful than losing it.
- Describe what problem could be introduced by a system that requires every single piece of data to be acknowledged and retransmitted before continuing.
- Relate this problem to the concept of overhead.
- Explain why, in this case, "unreliable" does not necessarily mean "poorly designed service."

Transferring a File Without Errors

A student needs to upload a 2 GB project file to the cloud.

During the transfer, some packets are lost because of temporary network problems. However, the system must guarantee that the file received by the server is identical to the original.

Answer the following questions:

- Explain why a reliable service is necessary in this situation.
- Explain what problem would occur if even a small part of the file were not received correctly.
- Describe the function of an acknowledgment.
- Explain what the sender should do if it does not receive confirmation for certain data.
- Explain why checks, acknowledgments, and retransmissions introduce additional overhead.
- Explain why, in this case, the additional overhead is still acceptable compared with losing data.
- Finally, compare this situation with a video call and explain why the same reliability strategy is not necessarily optimal for both.

Designing a Location Tracking System

An application shows the real-time position of a bus on a map.
Every three seconds, the bus sends its current position:

- 10:00:00 → Square A.
- 10:00:03 → Street B.
- 10:00:06 → Street C.
- 10:00:09 → Street D.

The 10:00:03 update is lost during transmission.
The designer must decide whether to retransmit it or continue with the following updates.

- Explain why each position update can be considered a message that is relatively independent of the others.
- Determine whether this service can be modeled as connection-oriented or connectionless according to the example in the learning material.
- Determine whether it can be considered reliable or unreliable.
- Explain why retransmitting the 10:00:03 position at 10:00:07 might be of little use.
- Explain why receiving the 10:00:06 position quickly may be more important than recovering the previous one.
- Imagine instead that the application must report that a package has been delivered and that this message cannot be lost. Explain why, in this new case, it would be appropriate to change the reliability requirement.
- Use the two cases to explain why a network service designer must evaluate the meaning and time sensitivity of the data before deciding whether to use acknowledgments and retransmissions.

Bibliography

Eddy, W. (Ed.). (2022). *Transmission Control Protocol (TCP)* (RFC 9293). RFC Editor. https://doi.org/10.17487/RFC9293

Eggert, L., Fairhurst, G., & Shepherd, G. (2017). *UDP usage guidelines* (RFC 8085). RFC Editor. https://doi.org/10.17487/RFC8085

Fairhurst, G., Trammell, B., & Kuehlewind, M. (Eds.). (2017). *Services provided by IETF transport protocols and congestion control mechanisms* (RFC 8095). RFC Editor. https://doi.org/10.17487/RFC8095

Postel, J. (1980). *User Datagram Protocol* (RFC 768). RFC Editor. https://doi.org/10.17487/RFC768

Schulzrinne, H., Casner, S., Frederick, R., & Jacobson, V. (2003). *RTP: A transport protocol for real-time applications* (RFC 3550). RFC Editor. https://doi.org/10.17487/RFC3550

Thomson, M., Damaggio, E., & Raymor, B. (2016). *Generic event delivery using HTTP push* (RFC 8030). RFC Editor. https://doi.org/10.17487/RFC8030

Further reading

- James F. Kurose and Keith W. Ross, Computer Networking: A Top-Down Approach, 9th ed., Pearson, 2026 — Provides a particularly clear introduction to transport-layer services, comparing connectionless UDP with connection-oriented TCP and explaining reliable data transfer, acknowledgments, retransmissions, connection management, flow control, congestion control, and socket programming.
- Andrew S. Tanenbaum, Nick Feamster, and David J. Wetherall, Computer Networks, 6th ed., Pearson, 2022 — Offers a systematic treatment of transport services and protocols, including connection-oriented and connectionless communication, UDP, TCP, QUIC, congestion control, performance, and streaming audio and video.
- Behrouz A. Forouzan, Data Communications and Networking with TCP/IP Protocol Suite, 6th ed., McGraw Hill, 2022 — Explains networking through the TCP/IP architecture and dedicates specific chapters to the transport layer and multimedia, making it useful for understanding how reliability, transport protocols, and real-time communication fit into the complete network stack.
- Kevin R. Fall and W. Richard Stevens, TCP/IP Illustrated, Volume 1: The Protocols, 2nd ed., Addison-Wesley Professional, 2012 — Provides a detailed examination of TCP/IP protocols, with particular attention to TCP connection establishment and termination, timeouts, acknowledgments, retransmissions, selective acknowledgments, packet reordering, and the mechanisms used to provide reliable communication.
- Douglas E. Comer, Internetworking with TCP/IP, Volume One, 6th ed., Pearson, 2014 — Explores the TCP/IP architecture in depth and provides a detailed treatment of reliable stream transport, explaining acknowledgments, retransmissions, timeouts, connections, sequence numbers, flow control, congestion, SACK, and ECN.

Editorial notes

In producing these materials, I have used generative artificial intelligence tools to support the writing process, particularly to improve the form of the text, reorganize content, refine wording, and speed up certain editorial tasks.

As I work independently on the production of these materials, I try to automate any activities that can reasonably be automated, so that I can devote more time to research, design, and content development.

However, artificial intelligence does not determine the content of this work: the choice of topics, structure, ideas, interpretations, examples, and teaching approach are developed by me. AI is therefore used primarily as a tool to support production and formal revision, while authorship and responsibility for the design and content remain mine.

Unless otherwise indicated, this material is distributed under the Creative Commons Attribution–NonCommercial–ShareAlike 4.0 International license (CC BY-NC-SA 4.0).

You are therefore free to share, redistribute, adapt, and create derivative works based on this material, provided that appropriate credit is given, the material is not used for commercial purposes, and any modified or derivative versions are distributed under the same license.

