

情報学基礎 第八回

5章 インターネット1: ネットワークの基礎

管理工学科

担当: 篠沢 佳久

本日の内容

- インターネット1: ネットワークの基礎(5章)
 - インターネットの構造(5.1節)
 - プロトコルの階層化(5.2節)
 - 物理層とデータリンク層プロトコル(5.3節)
 - ネットワーク層プロトコル(5.4節)
 - トランスポート層プロトコル(5.5節)

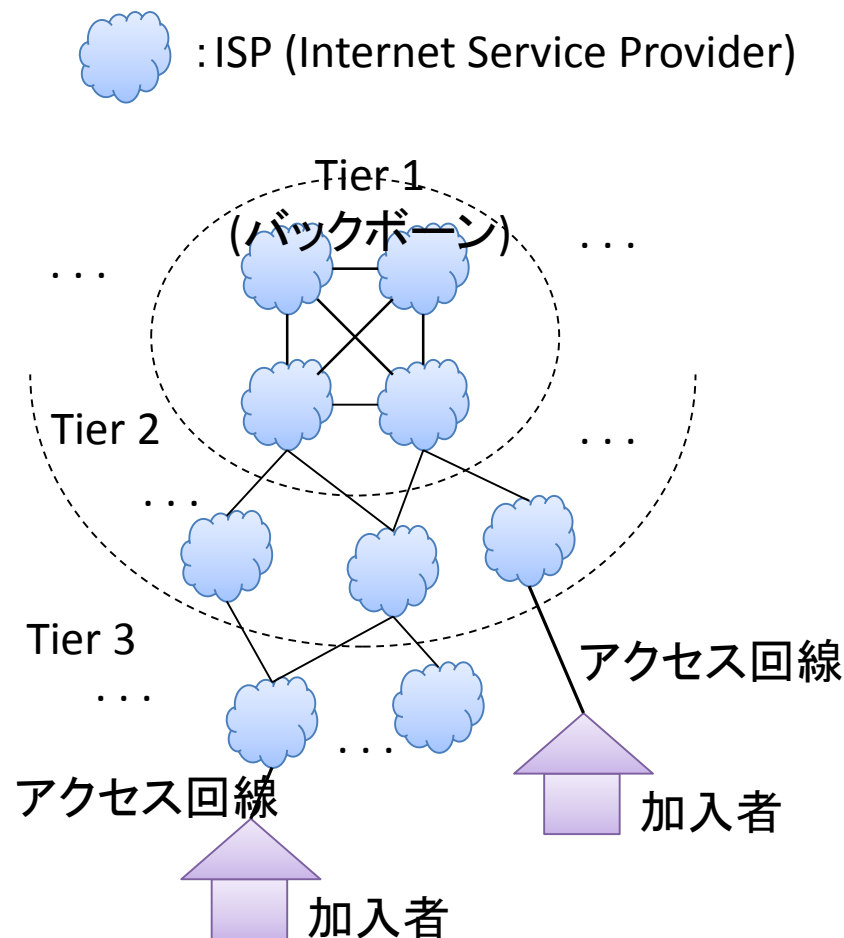
インターネットと歴史

- インターネットはさまざまなデジタル情報を転送する汎用的なネットワーク
 - メール, 文章, 音声, 動画など.
- 1970年代: 米国で基本方式が提案される
 - 電話の発明は1876年
- 80年代初頭: 大学や研究所をつなぐ研究用ネットワークとして運用開始
- 80年代後半: 米国で商用インターネット接続業者 (ISP: Internet Service Provider) が登場
 - 日本では1992年に最初のISPがサービスを開始
- 90年代中頃から世界中で爆発的に普及

インターネットの構造(5.1節)

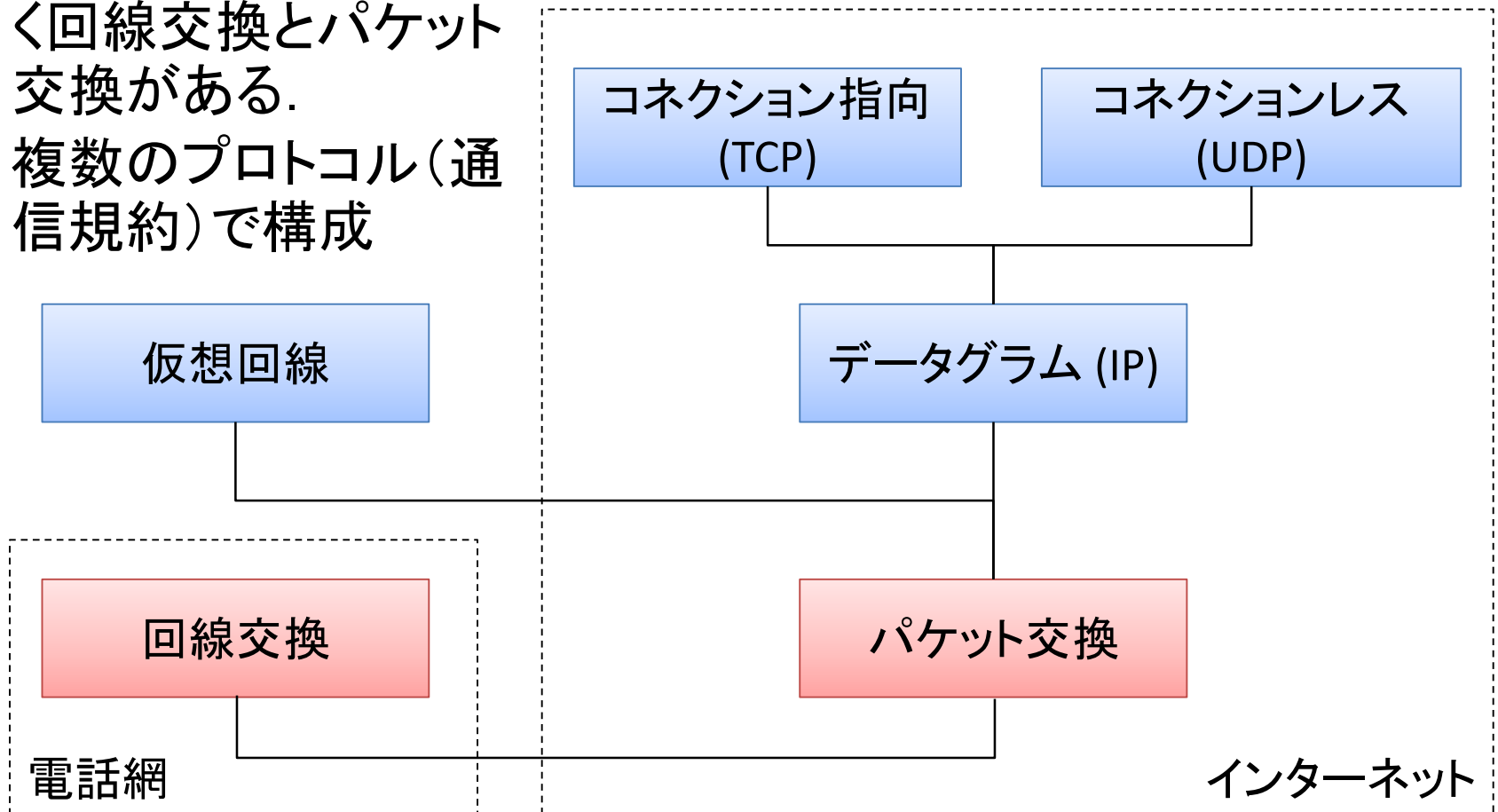
インターネットとは?

- TCP/IPと呼ばれる**プロトコル**群を使用
 - プロトコル = 通信規約
- インターネット = ISPの集合
 - **network of networks**
 - 階層構造をとる
 - tier 1, tier 2, tier 3, ...
 - ISPと加入者(家庭や企業)は**アクセス回線**を通じて接続
- インターネット = ASの集合
 - **AS: Autonomous System**
 - 同一の運用方針の範囲
 - 1つのISP or 複数ISPの集合



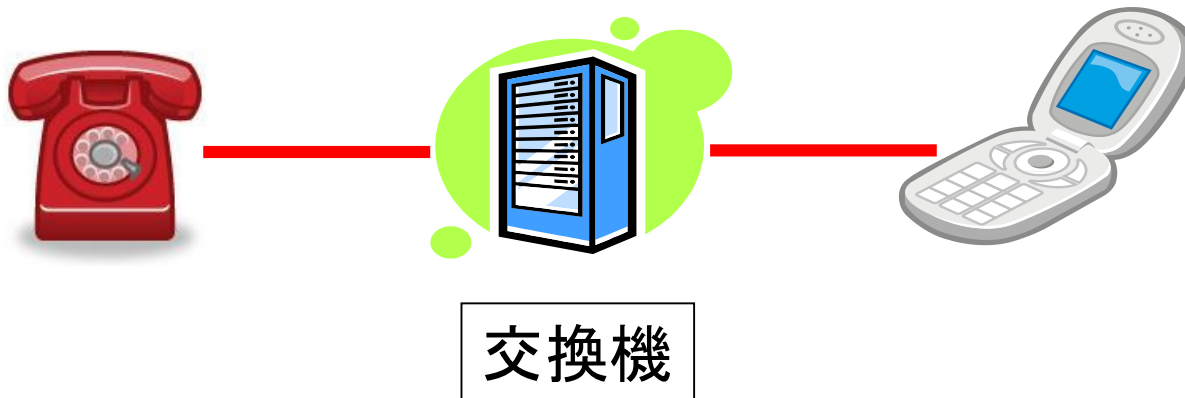
通信方式の分類

- 通信方式として, 大きく回線交換とパケット交換がある.
- 複数のプロトコル(通信規約)で構成



回線交換とパケット交換①

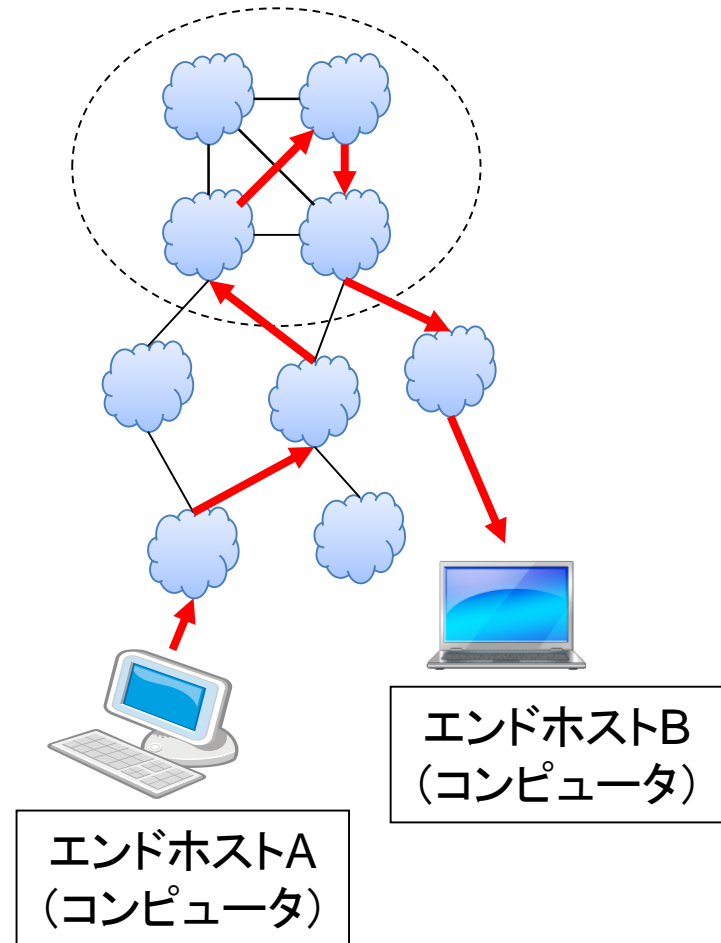
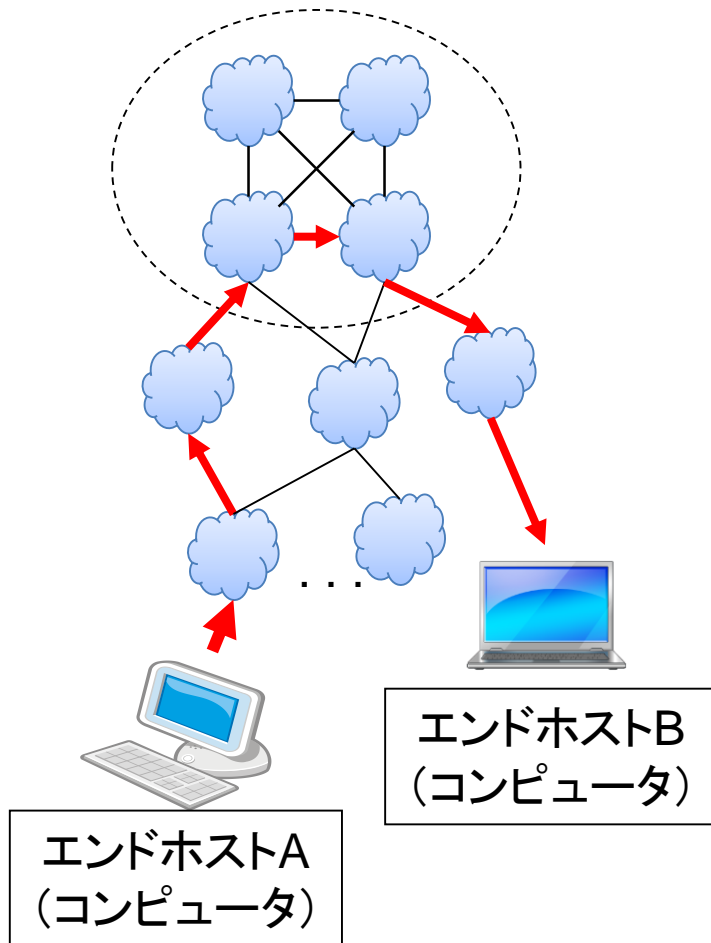
- 回線交換（電話網）
 - 電話網: 交換機が相互に接続
 - 通信（通話）開始前に「回線」を確立し, 占有
 - 通信品質保証が容易
 - 同時に行なえる通信の数はパケット交換より少ない



回線交換とパケット交換②

- **パケット交換**（インターネットはこちら）
 - インターネット: **ルータ**が相互に接続
 - インターネットに接続する機器: **ホスト / エンドホスト**
 - データを “**パケット**” に分割して送信, ネットワーク資源を**共有**
 - 通信品質保証は困難
 - 通信の収容数は回線交換より多い

パケット交換①



パケット交換②

送りたい情報(メッセージ)

1. パケットに分割

パケット1 パケット2 ... パケットn

2. 様々な送信処理

処理1
処理2
処理3
処理4
....



エンドホストA
(コンピュータ)



ルータ
(中継装置)



エンドホストB
(コンピュータ)

パケットn ... パケット2 パケット1



3. パケットごとに送信・伝送

パケット交換③

受信した情報(メッセージ)

6. パケットを集めて組み立て

パケット1

パケット2

...

パケットn



エンドホストA
(コンピュータ)



ルータ
(中継装置)



エンドホストB
(コンピュータ)

5. 再送
などの処理

パケットn

...

パケット2

パケット1

1はOK!

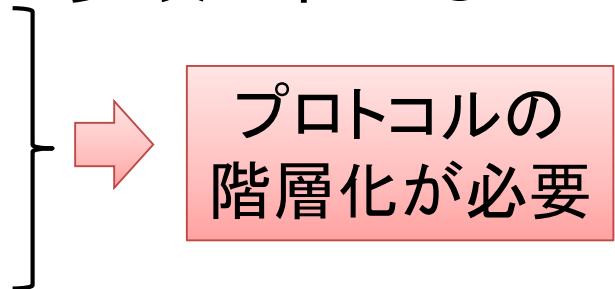
2を再送!

4. 応答(ACK)

様々な通信規約 = プロトコル

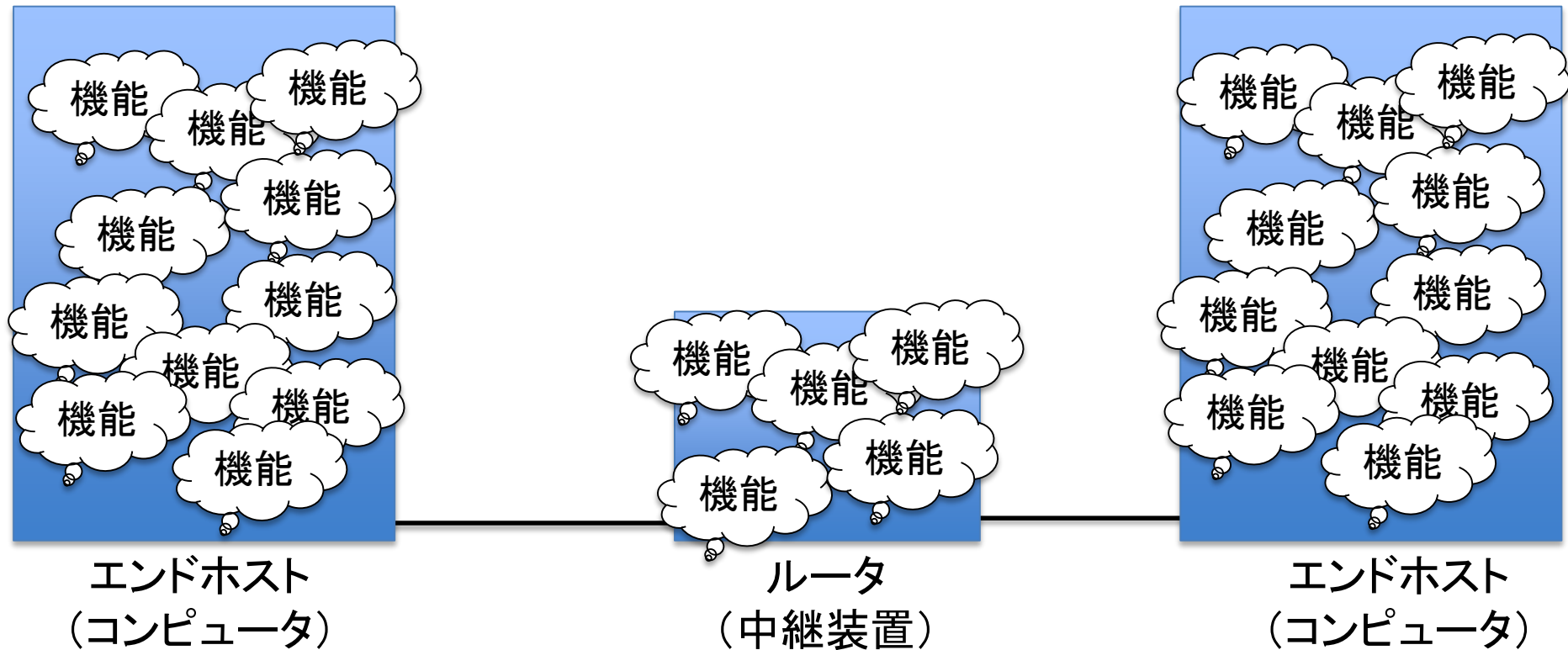
プロトコルの階層化(5.2節)

プロトコルの階層化 (1)

- **プロトコル**(protocol): 通信における決まり事 (通信規約)
 - パケットフォーマット, パケット送受信の手順などを定義
- 物理的な規約から意味的な規約まで多岐にわたる
 - コネクタの形状, 電圧, 符号化
 - 経路の選択
 - パケット損失の検出方法, 回復方法
 - 情報の意味, etc.

プロトコルの階層化が必要
- OSI参照モデル: 7階層
 - OSI: Open Systems Interconnection
- **インターネットの階層モデル**: 5階層

多種多様な機能を整理したい → モデル化



プロトコルの階層化: インターネットの5階層モデル



プロトコルの階層化: 各階層の機能

- アプリケーション層

- アプリケーションごとのデータの交換や処理
- 例: 電子メール (SMTP), Web (HTTP), etc.

- トランスポート層

- アプリケーションプロセス間のデータ転送
- 例: TCP, UDP, etc.

- ネットワーク層

- エンドホスト間のデータ転送
- 例: IPv4, IPv6, etc.

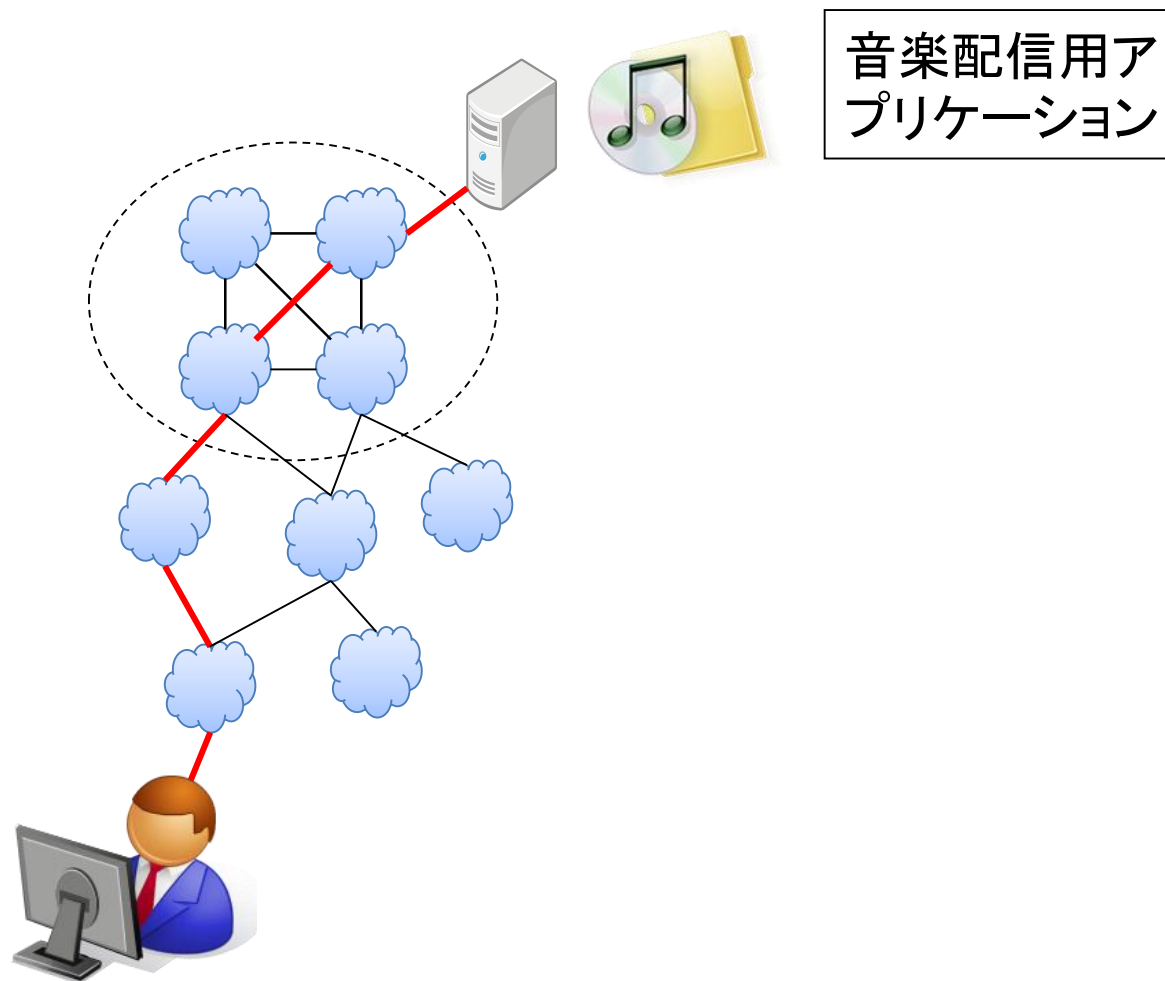
- データリンク層

- 対向するノード間でのデータ転送
- 例: Ethernet, etc.

- 物理層

- 対向するノード間での信号の送受

プロトコルの階層化: 送受信データの流れ



音楽配信用アプリケーション

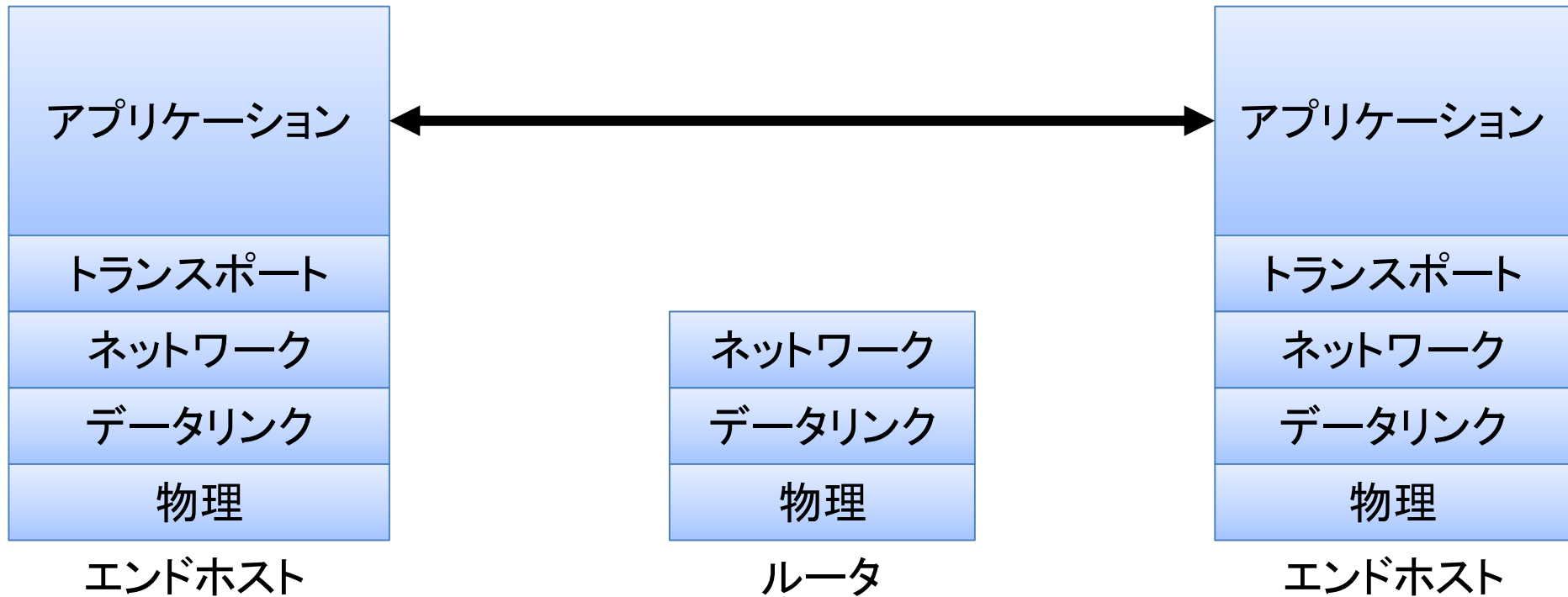
音楽ダウンロード
アプリケーション

プロトコルの階層化:送受信データの流れ



音楽配信用アプリケーション

音楽ダウンロードアプリケーション



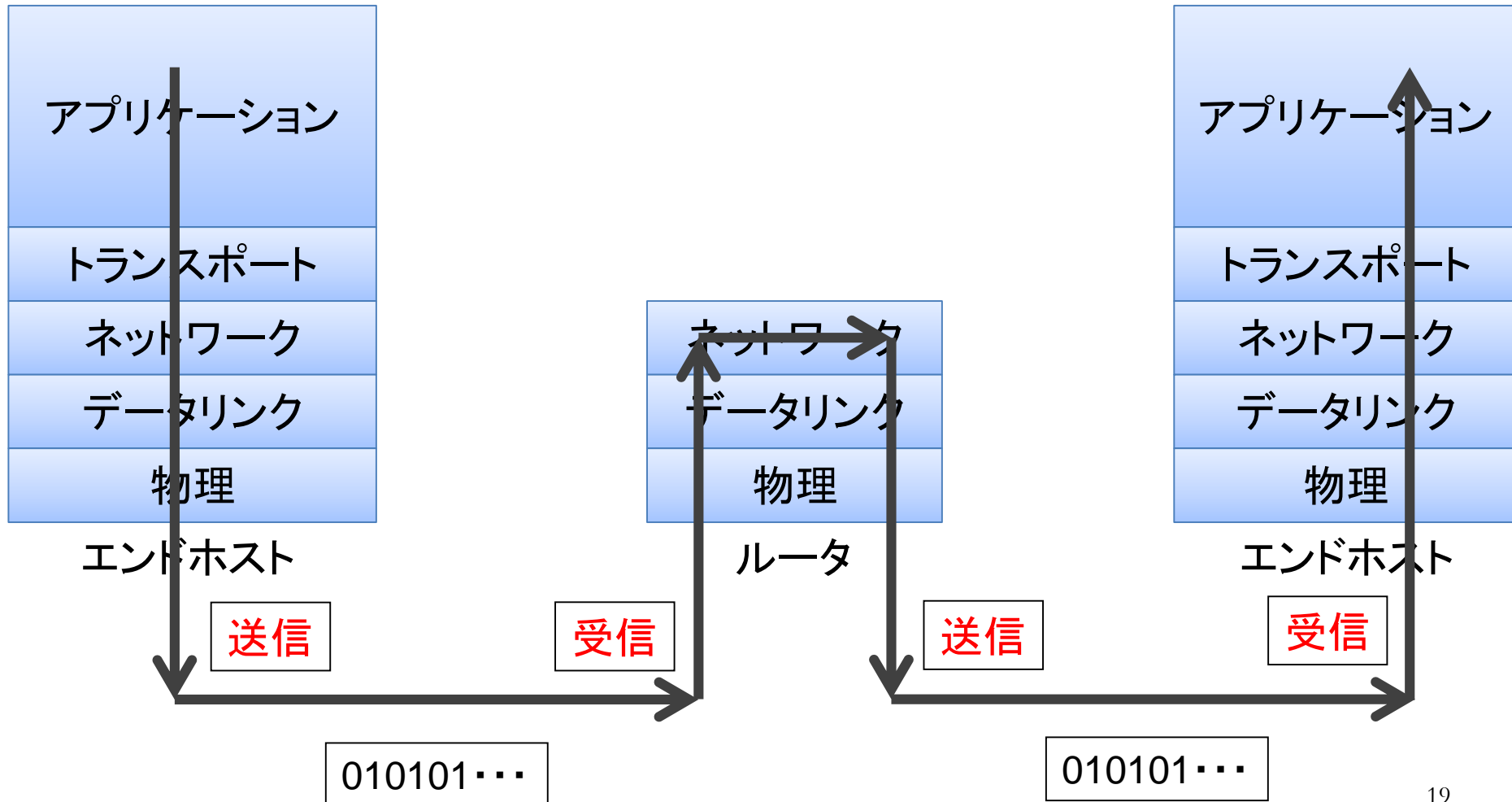
このように通信しているように見えますが...

プロトコルの階層化: 送受信データの流れ



音楽配信用アプリケーション

音楽ダウンロードアプリケーション



プロトコルの階層化: 送受信データの流れ

- 送信

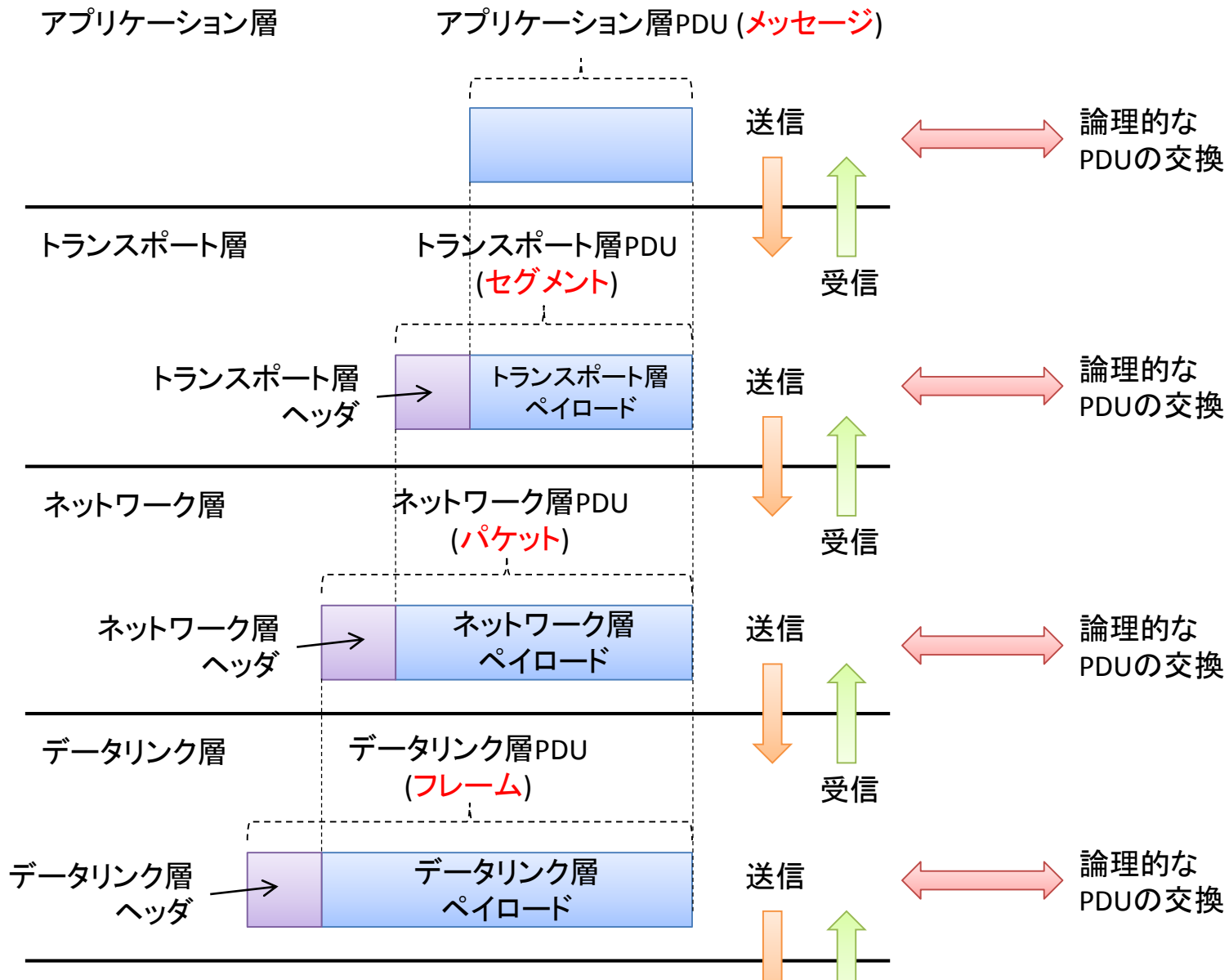
- (複数の)上位層から受け取ったペイロードにヘッダを付加してPDUとし, 下位層へ → **多重化** (multiplexing)
- **PDU** (Protocol Data Unit) = **ヘッダ** + **ペイロード**
 - ある階層のPDU = 1つ下の層のペイロード
 - 例: トランスポート層PDU = ネットワーク層ペイロード

- 受信

- 下位層から受け取ったPDUからヘッダを削除したペイロードを(複数のうちの1つの)上位層へ → **逆多重化** (demultiplexing)

- 論理的には各階層が直接PDUを交換していると考える

プロトコルの階層化: 送受信データの流れ



プロトコルの階層化: 利点

- 機能詳細の隠蔽

- ある階層は下位層が提供するサービスがどのように実現されているかを気にする必要はない
- 例: アプリケーション層設計者は, トランスポート層サービス (e.g., 信頼性保証通信) の詳細を気にすることなく利用できる

- モジュール性の高さ

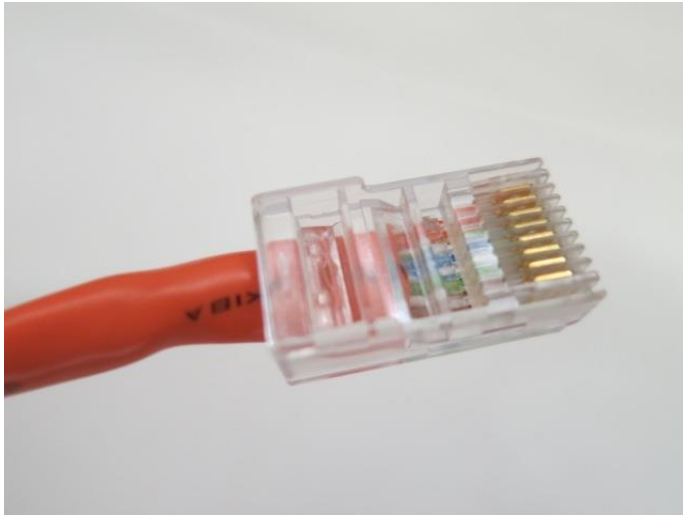
- 階層間の取り決めを守っていれば, サービスの実現方法を自由に変えられる
- 例: トランスポート層での損失パケット回復方法の改良

物理層とデータリンク層プロトコル(5.3節)

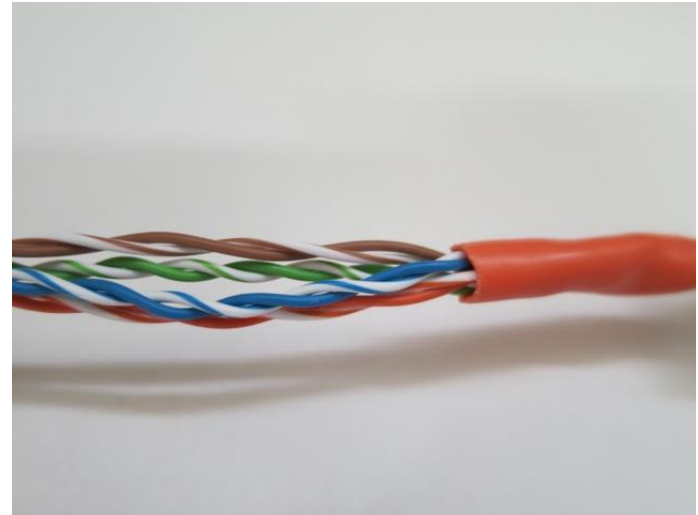
物理層(5.3節)

- 物理層は, 通信媒体を使って物理的な信号を送受することで, ビットの伝送を実現する
- 通信媒体:
 - 有線: 撚り線対(よりせんつい), 同軸ケーブル, 光ファイバなど
 - 無線: 電波, 赤外線, 可視光など
- 通信のための電気的特性や物理的特性を規定
 - 電気的特性: 信号方式, 符号化方式
 - ビットを物理信号でどのように表現するか
 - 物理的特性: コネクタの形状など

通信媒体の例：撚り線対



(a) コネクタ部分



(b) 被覆の内側

図5-5:イーサネットケーブルとRJ-45コネクタ

マンチェスタ符号

- 電圧の変化でビット(1 か 0 か)を表す.
 - 高電位→低電位は0, 低電位→高電位は1を表す.

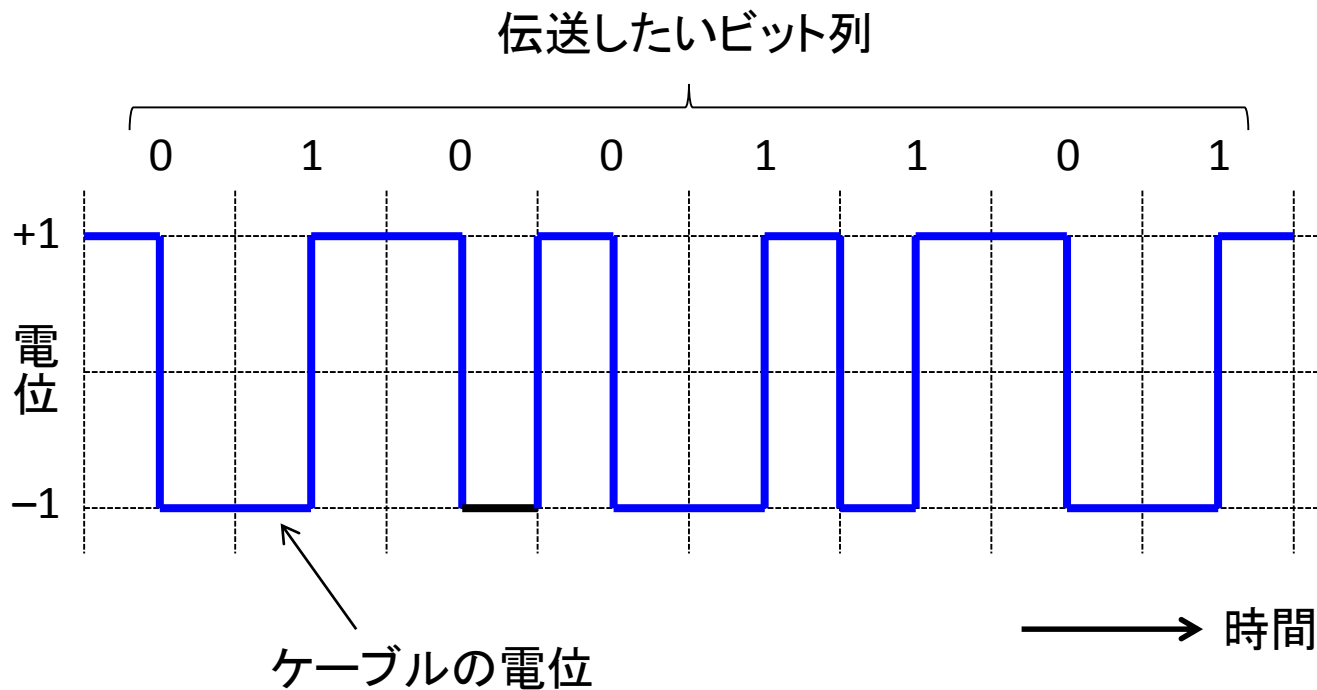


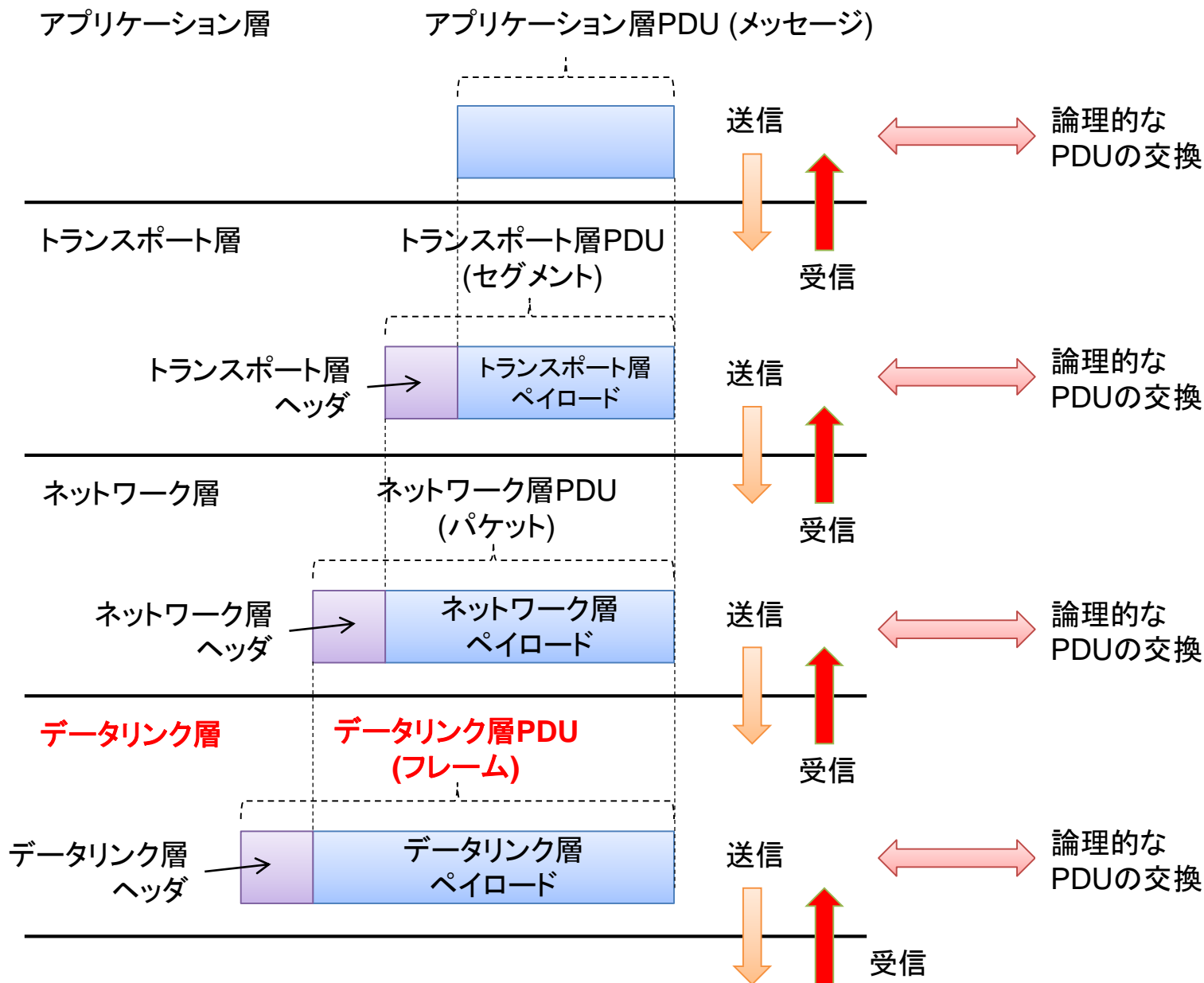
図5-6: マンチェスタ符号の例

伝送速度

- ネットワークの速度は伝送速度で表現されることが多い。
 - 例: 100Mbps のインターネット接続
- 伝送速度は「1秒間に伝送できるビット数」:
bps; bit per second
 - 1秒間に1ビット伝送: 1bps
 - 1秒間に1,000 ビット伝送: 1000 bps = 1kbps
 - 1秒間に1,000,000 ビット伝送: 10^6 bps = 1Mbps
 - 1秒間に1,000,000,000ビット伝送: 10^9 bps = 1Gbps
- r bps で n bit のデータを伝送するのに必要な時間 t
$$t = n / r \quad (1) \quad [\text{bit}] / ([\text{bit}] / [\text{sec}]) \Rightarrow [\text{sec}]$$
- ユーザから見た伝送速度はもっと遅くなる.

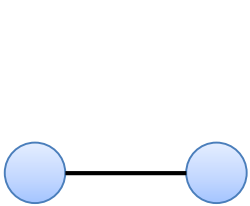
伝送速度は,
○ 1k = 1000
× 1k = 1024

プロトコルの階層化: 送受信データの流れ

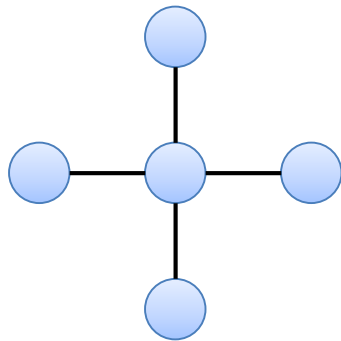


データリンク層の役割

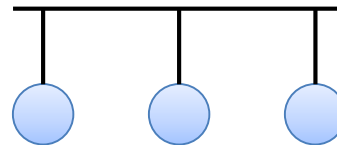
- 1つの通信媒体で接続されたホスト間の通信機能を提供
- さまざまな接続形態



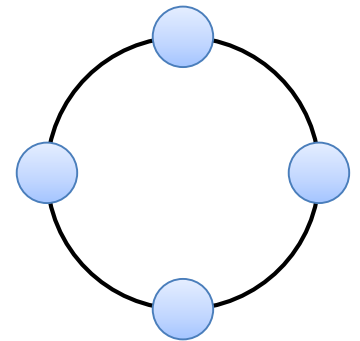
(a) 対向接続



(b) スター接続



(c) バス接続

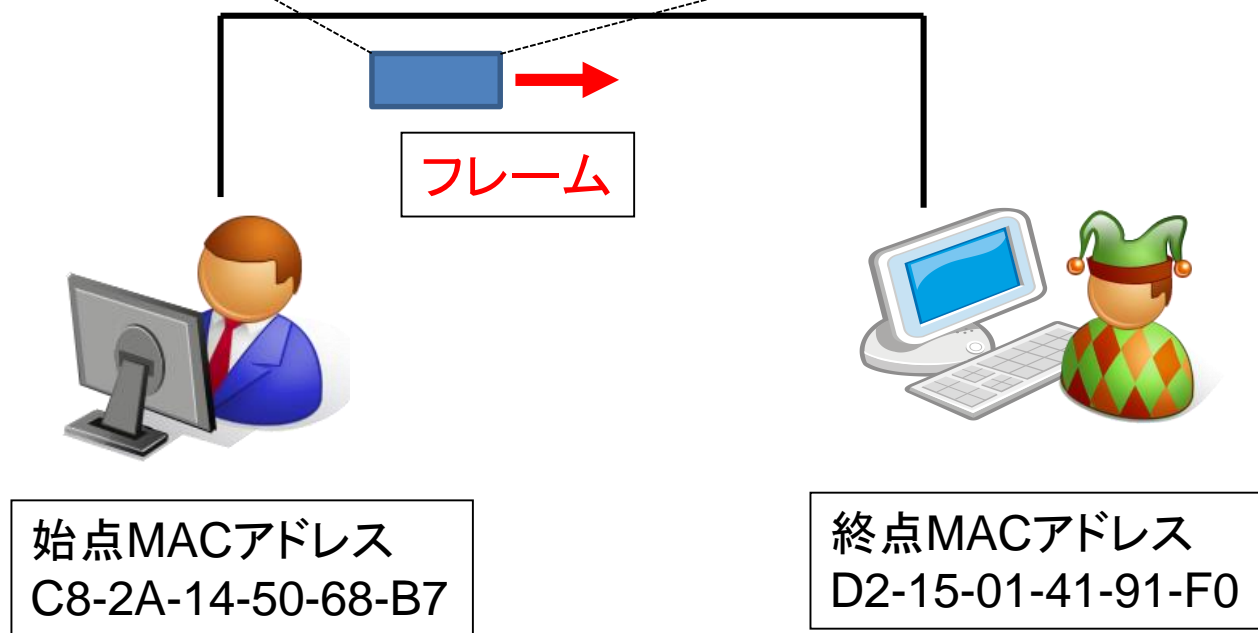
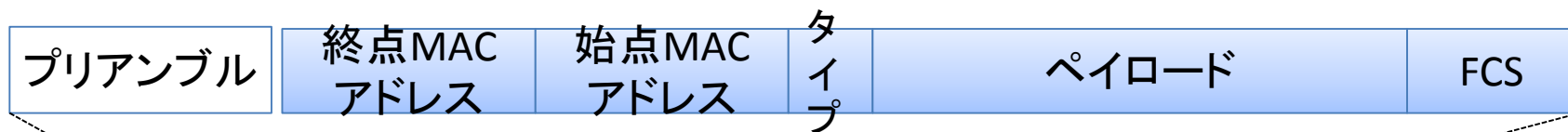


(d) リング接続

代表例: イーサネット (Ethernet)

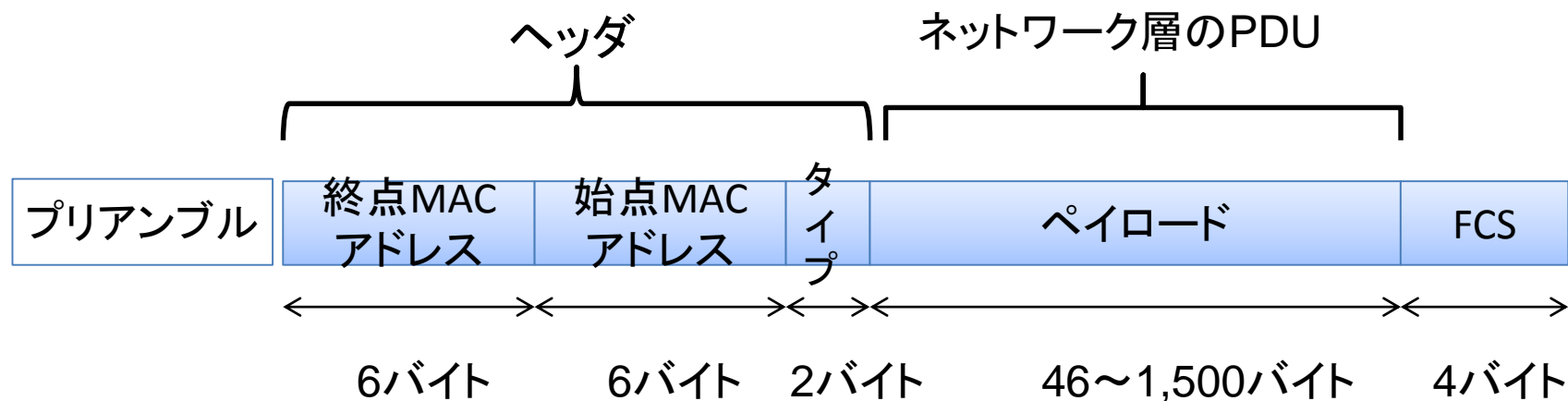
- 1976年に米国Xerox社が実用化
 - 同軸ケーブルを利用したバス接続, 通信速度は3 Mbps
 - bps: bit per second, Mbps = 10^6 bps
 - 1982年にDIX ver.2 と呼ばれる規格となる
- 一方, 1985年にIEEEはIEEE802.3という国際標準にした
 - IEEE: Institute of Electrical and Electronics Engineers
 - 当初10 Mbps, その後, 100 Mbps, 1 Gbps, 10 Gbps と高速化
 - 媒体は同軸ケーブルから撚り線対, 光ファイバへ
 - 接続形態はバス接続から, スイッチを中心としたスター接続へ

イーサネットフレーム (5.3.3節)



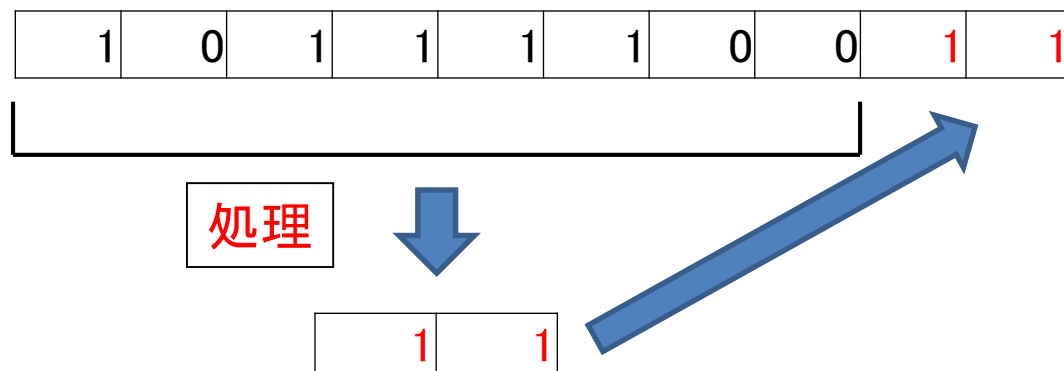
イーサネットフレームフォーマット

- イーサネットではフレームの塊でデータを伝送
- プリアンブル: フレーム同期に使用
 - 伝送路を流れるビットの列からフレームを検出
- ヘッダ(先頭部分): 制御用の情報が格納
 - 終点MACアドレス(宛先)と始点MACアドレス(送信元)
- ペイロード: 伝送するデータが格納
 - ヘッダのタイプにペイロードのタイプが記述
- FCS(Frame Check Sequence): ビットの誤りを検出



ビット誤り検出(参考)

- CRC(Cyclic Redundancy Check)
- チェックサム

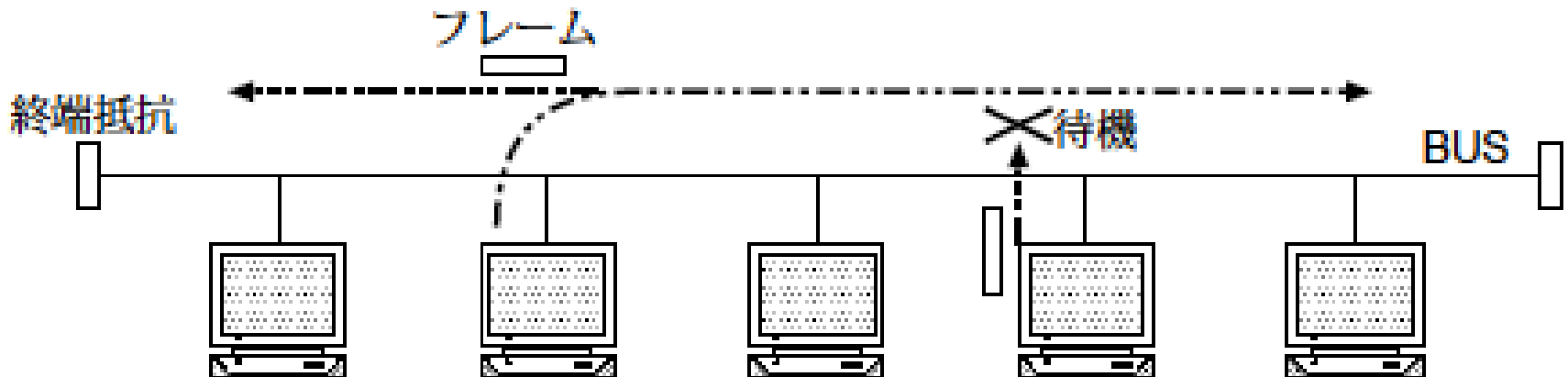


MACアドレス(5.3.1節)

- **MACアドレス**: データリンク層アドレス
 - データリンク層でのホストの識別子
 - **EUI-48**(48-bit Extended Unique Identifier)という番号体系
 - 48ビット(6バイト)
 - 先頭24ビット: **OUI**(Organizationally Unique Identifier)
 - ハードウェアメーカーに割り当てられた番号
 - 後半24ビット: ハードウェアメーカーが一意に割当
- 表記法
 - 8ビットごとの6つの16進数で表記
 - 例: C8-2A-14-50-68-B7 あるいは C8:2A:14:50:68:B7

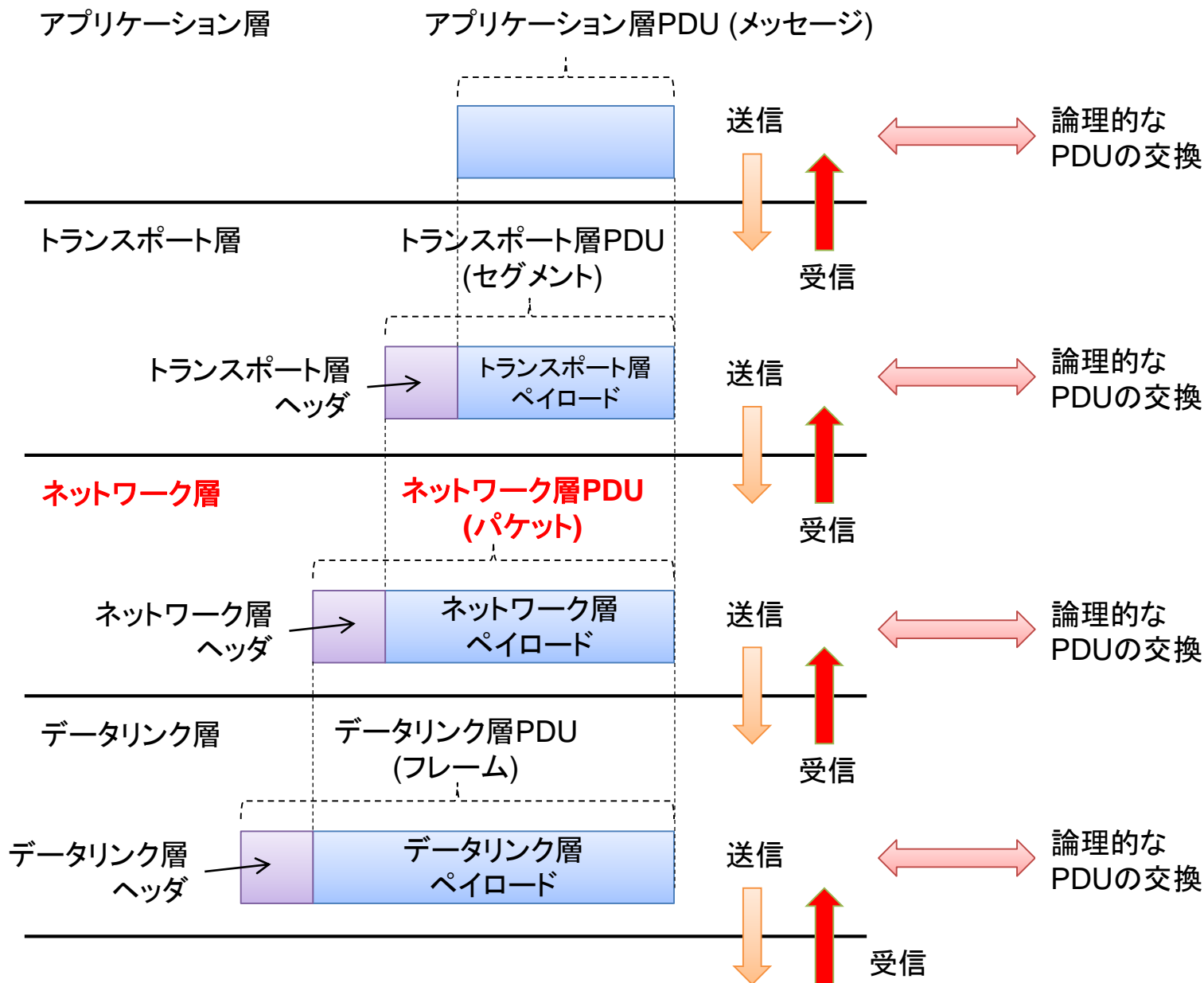
フレームの送信制御: CSMA/CD (5.3.2節)

- **CSMA/CD**: バス接続での送信制御方式の一種
- MA (multiple access): 複数ホストでひとつのバスを共用
 - バス接続では, 一度には1つのホストしか送信できない
 - 複数ホストが同時に送信 → 信号の衝突 (**コリジョン**)
- CS (Carrier Sense): 他の送信がないか確認
 - バスが空いていれば送信. いなければ待機.
- /CD (with Collision Detection): 衝突を検知したら送信中断
 - exponential back-off : ランダム時間待って再送.
 - 待ち時間はコリジョンのたびに指数関数的に増加



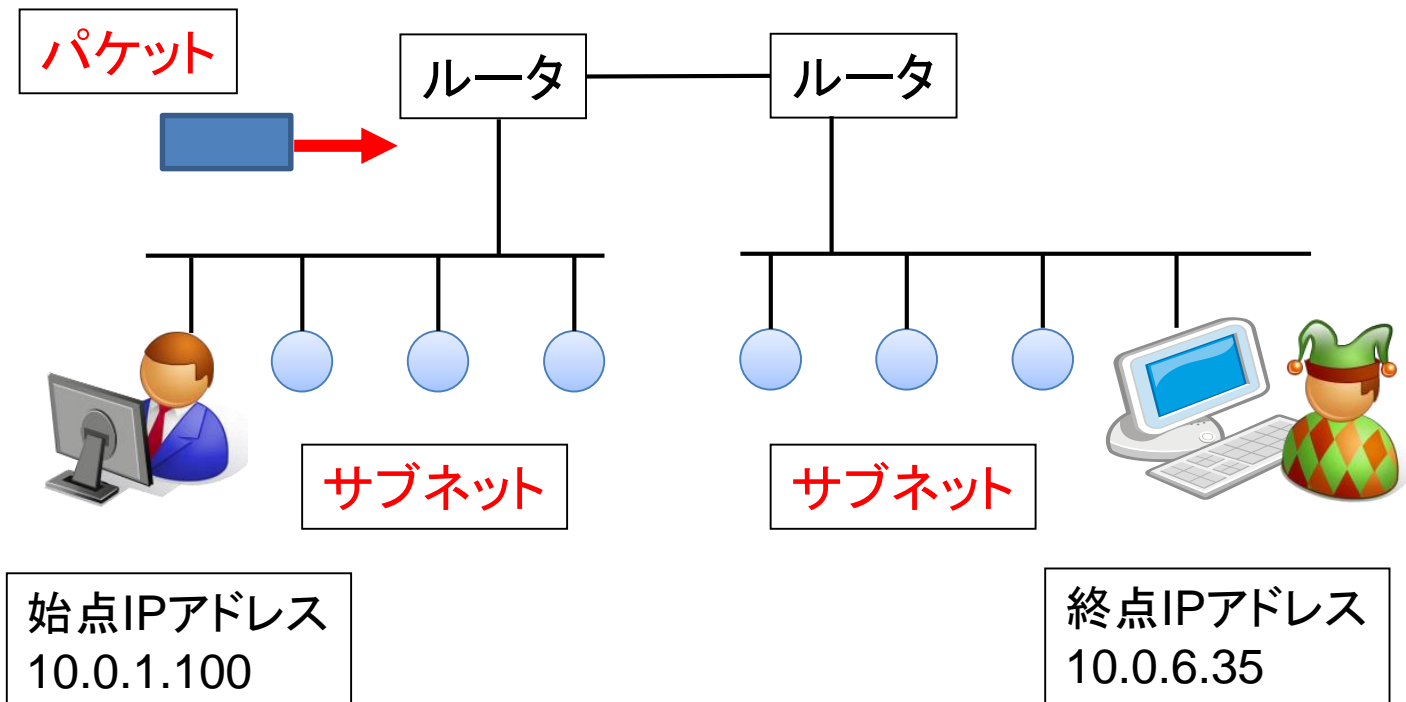
ネットワーク層プロトコル (5.4節)

プロトコルの階層化: 送受信データの流れ



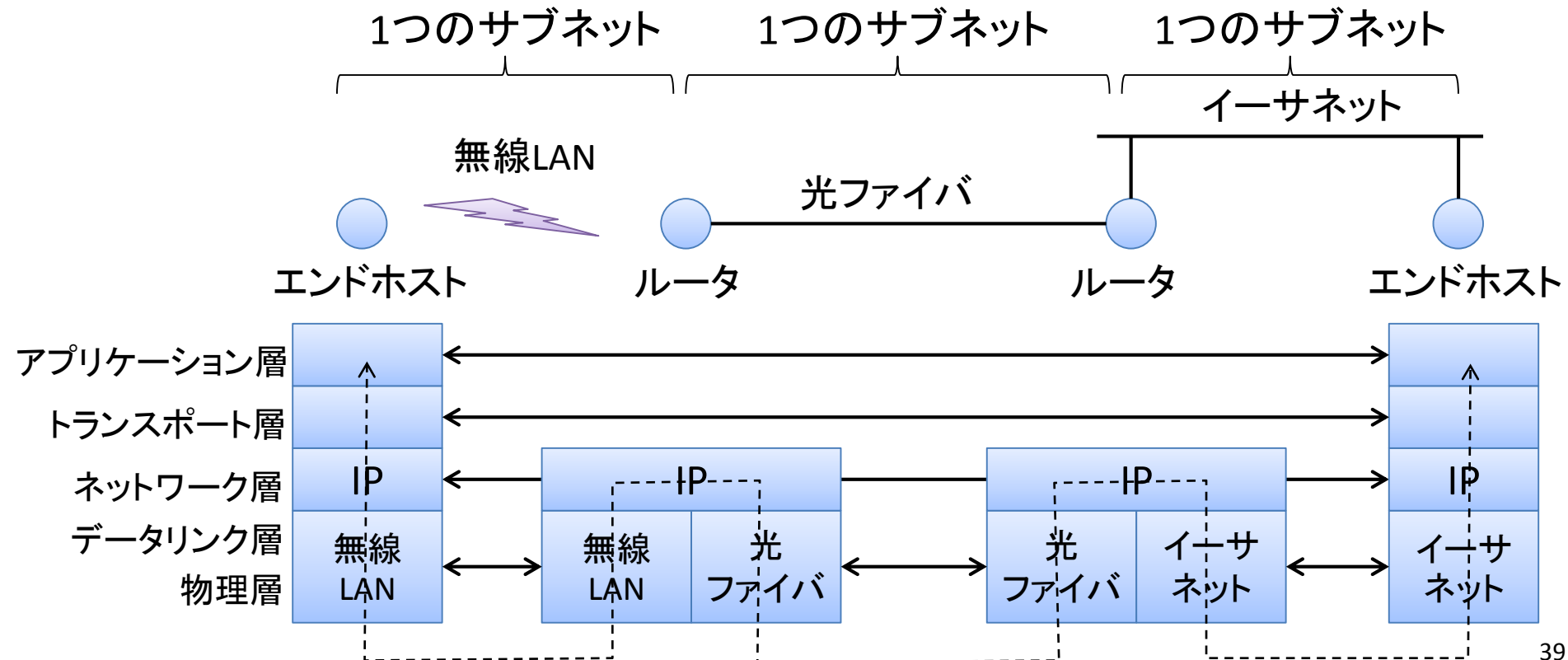
ネットワーク層の役割(5.4節)

- ルータを介したエンドホスト間での通信機能を提供
- 異種データリンクを IP (Internet Protocol) で相互接続
 - Internetworking with IP



ネットワーク層の役割

- ルータを介したエンドホスト間での通信機能を提供
- 異種データリンクを **IP (Internet Protocol)** で相互接続
 - Internetworking with IP



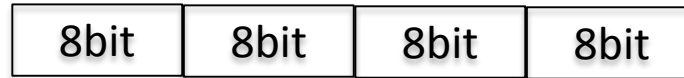
IPとIPアドレス

- **IPv4** (IP version 4): 現在主流のネットワーク層プロトコル
 - 32ビットのIPアドレス → すでに枯渇
 - $2^{32} \doteq 43$ 億
- **IPv6** (IP version 6): IPv4の後継プロトコル
 - 128ビットのIPアドレス → 実用上枯渇することはない
 - $2^{128} \doteq 3.4 \times 10^{38} = 340 \times 1兆 \times 1兆 \times 1兆$
- **IPアドレス**: ネットワーク層でのホストの接続場所を示す
 - cf. MACアドレスは1つの媒体でのホストの識別
 - IPアドレス = サブネット番号 + ホスト番号
 - サブネット: ルータを介さずに通信できる範囲, あるいは, 1つの通信媒体の範囲

IPv4アドレス

- インターネット上のホストを指定するためのアドレス
- 表記法: 8ビットごとの4つの10進数で表記

– 例: 131.113.71.3

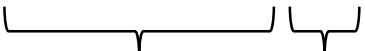


32bit

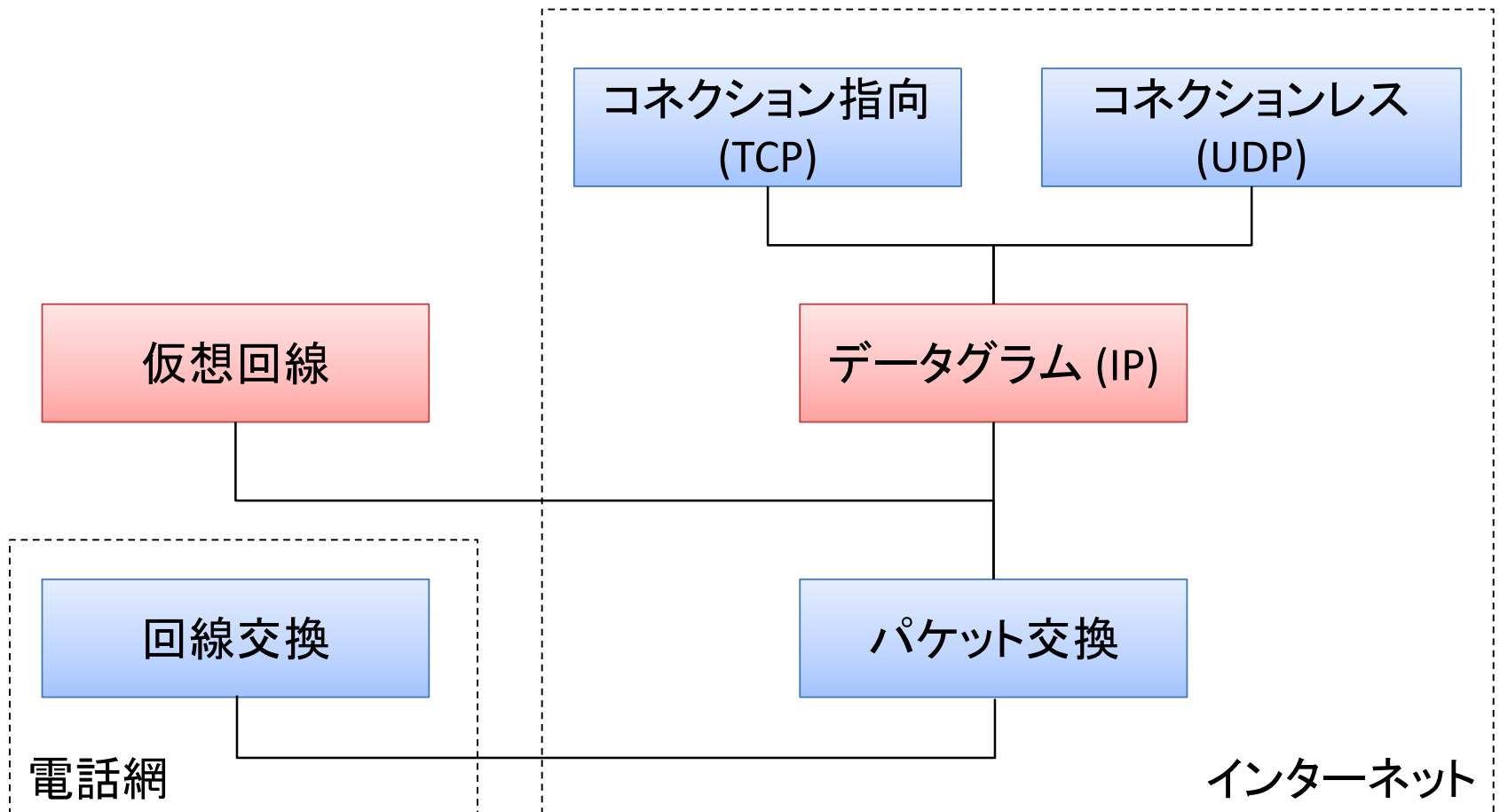
- IP アドレス = サブネット番号 + ホスト番号
 - かつては, アドレスの先頭から固定的な長さ x ビットはサブネット番号を表すなどと決めていた => 現在は可変

- **CIDR** (Classless Inter-Domain Routing)

- サブネット番号のビット長を “/” のあとに明示
- 例: 131.113.71.3/24 (先頭から24ビットがサブネット番号)


サブネット番号 ホスト番号

通信方式の分類 (再掲)



パケット転送方式(5.4.2節)

- 仮想回線

- 通信開始前に仮想回線を確立, 終了時には仮想回線を解放
 - ルータが仮想回線のための状態を保持 → 多数の通信の収容は困難
- パケットにどの仮想回線かを示す “タグ” をつけて送信
- 通信品質保証は容易

- データグラム (インターネットはこちら)

- 仮想回線の確立・解放は不要
 - ルータは状態を持たない → より多くの通信の収容が可能
- パケットに送信元(始点)と宛先(終点)のIPアドレスを格納
- 通信品質保証は困難 → ベストエフォート
- IP (Internet Protocol, 後述)

データグラム方式におけるパケット転送

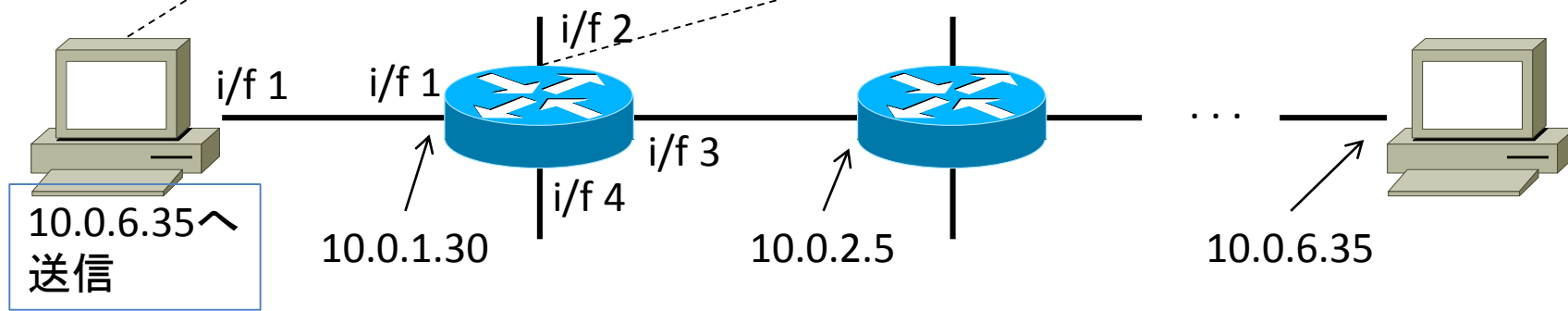
- 各ホストやルータは**経路表**を保持
- パケットの**終点アドレス**で経路表を検索
→ 次に転送すべきルータを決定
- 上記を始点ホストから終点ホストまで繰り返す

経路表

終点アドレス	次ホップ	出力先
10.0.6.0/24	10.0.1.30	i/f 1
...

経路表

終点アドレス	次ホップ	出力先
10.0.6.0/24	10.0.2.5	i/f 3
...



経路制御(5.4.6節)

経路表

終点アドレス	次ホップ
B	I

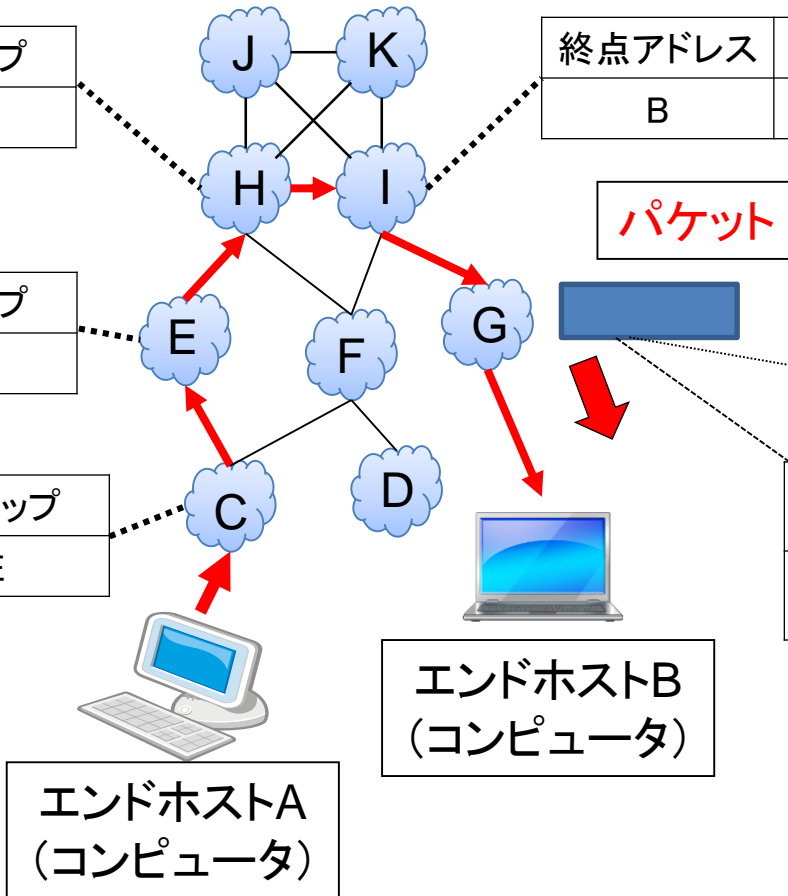
終点アドレス	次ホップ
B	H

終点アドレス	次ホップ
B	E

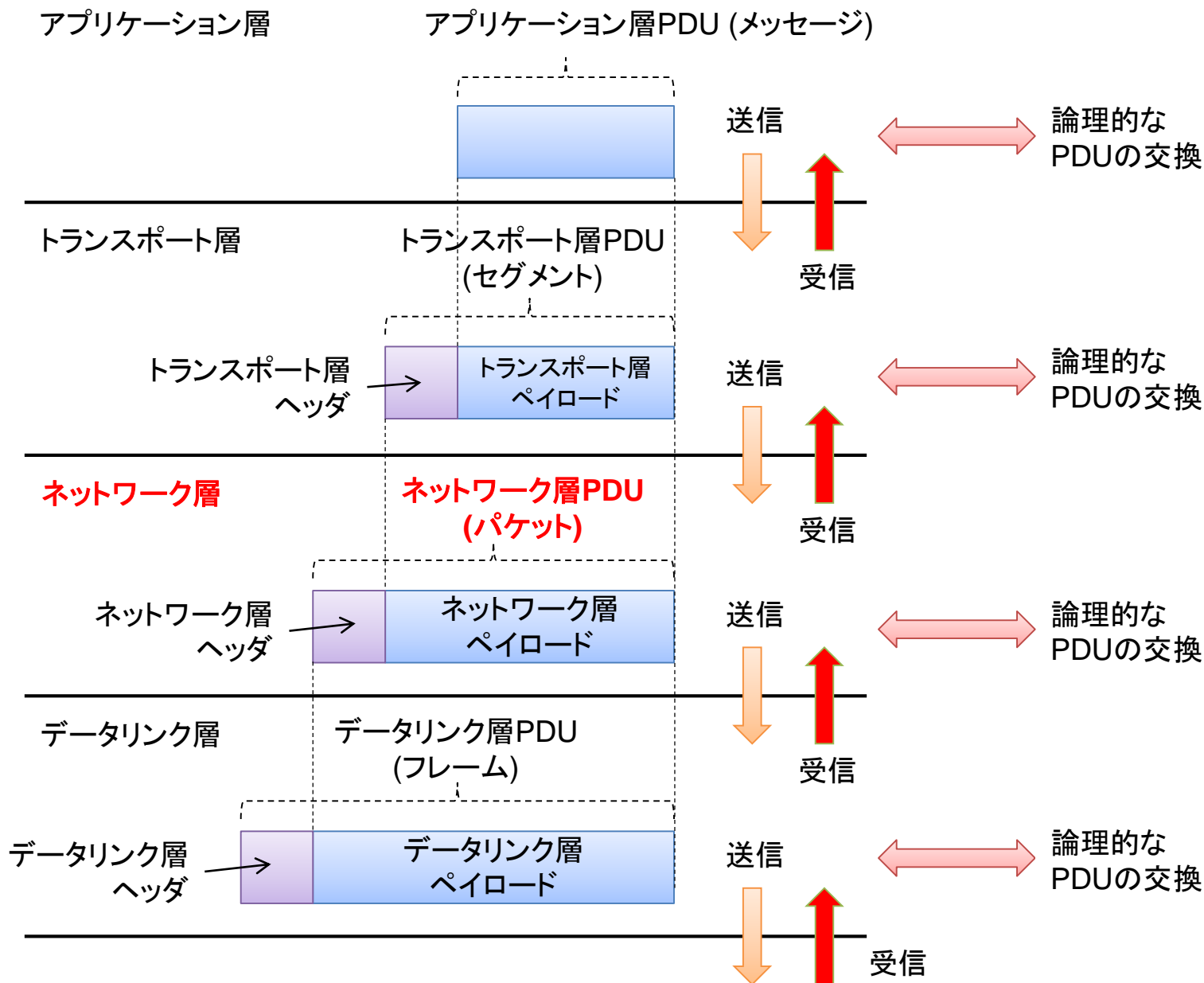
終点アドレス	次ホップ
B	G

パケット

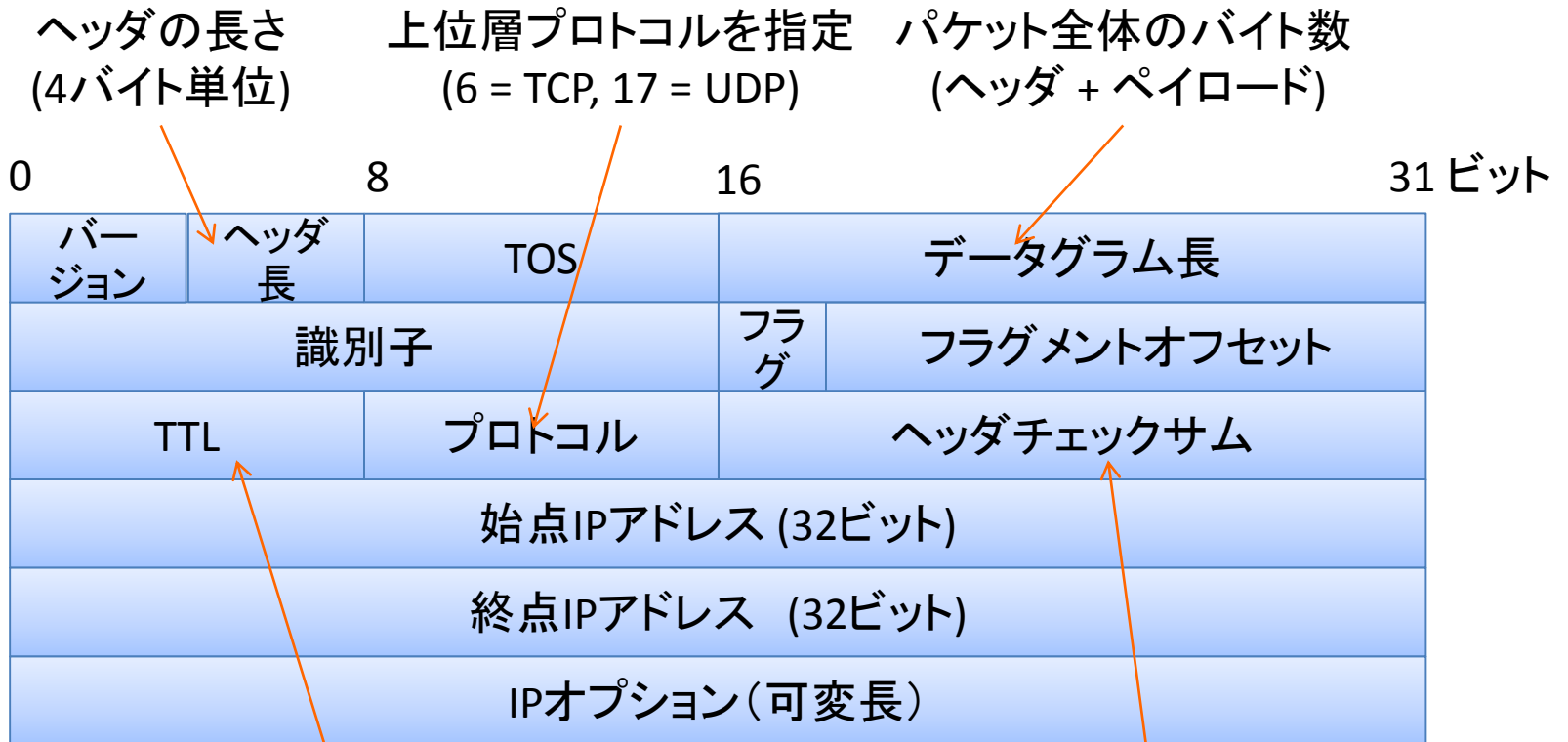
始点アドレス	終点アドレス
A	B



プロトコルの階層化: 送受信データの流れ



IPv4ヘッダ



ルータで転送のたびに1ずつ減算
→ 0になったらパケットを廃棄

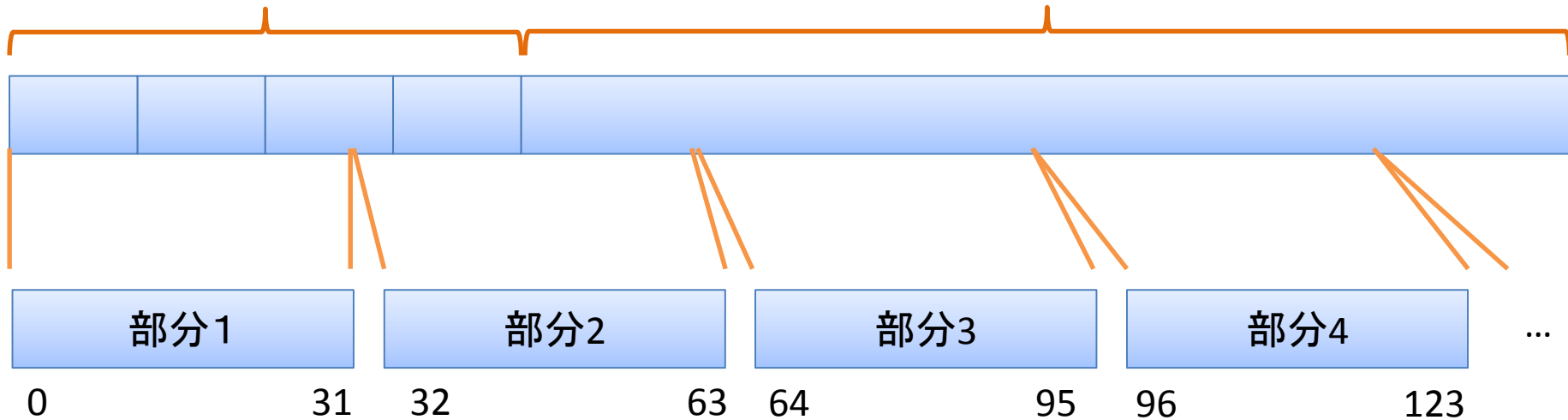
ヘッダのみの
チェックサム

図の見方

パケット ... 横に描くと、とても横に長くなる...

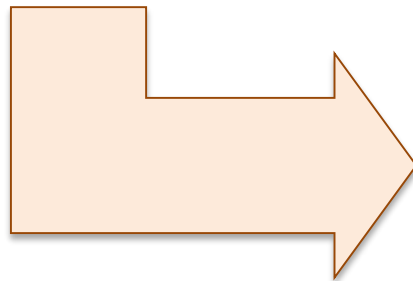
ヘッダ

ペイロード



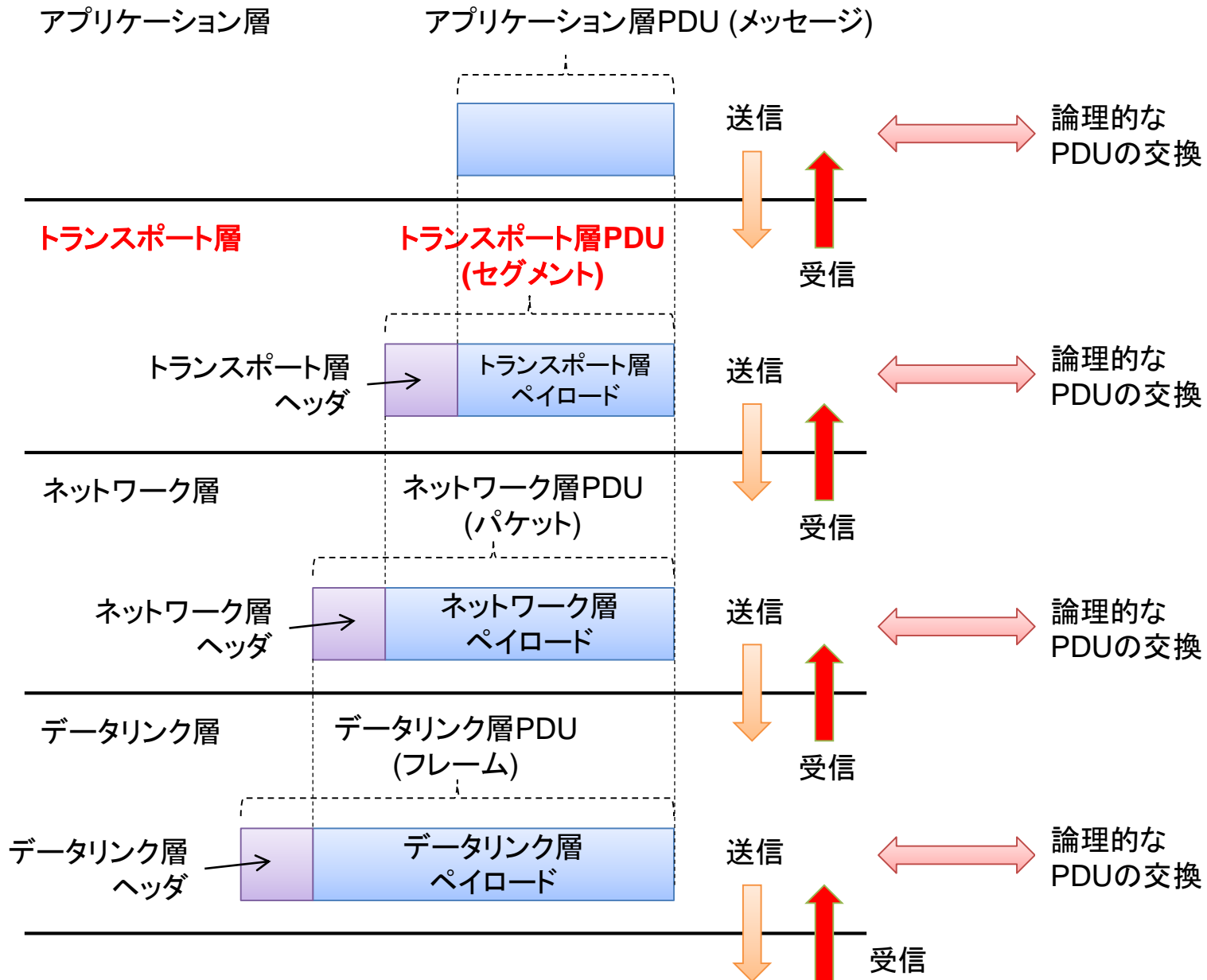
そこで、一定長に分割（例えば32ビットごと）

さらに、各部分を
縦に並べて描いて、
コンパクトに表現



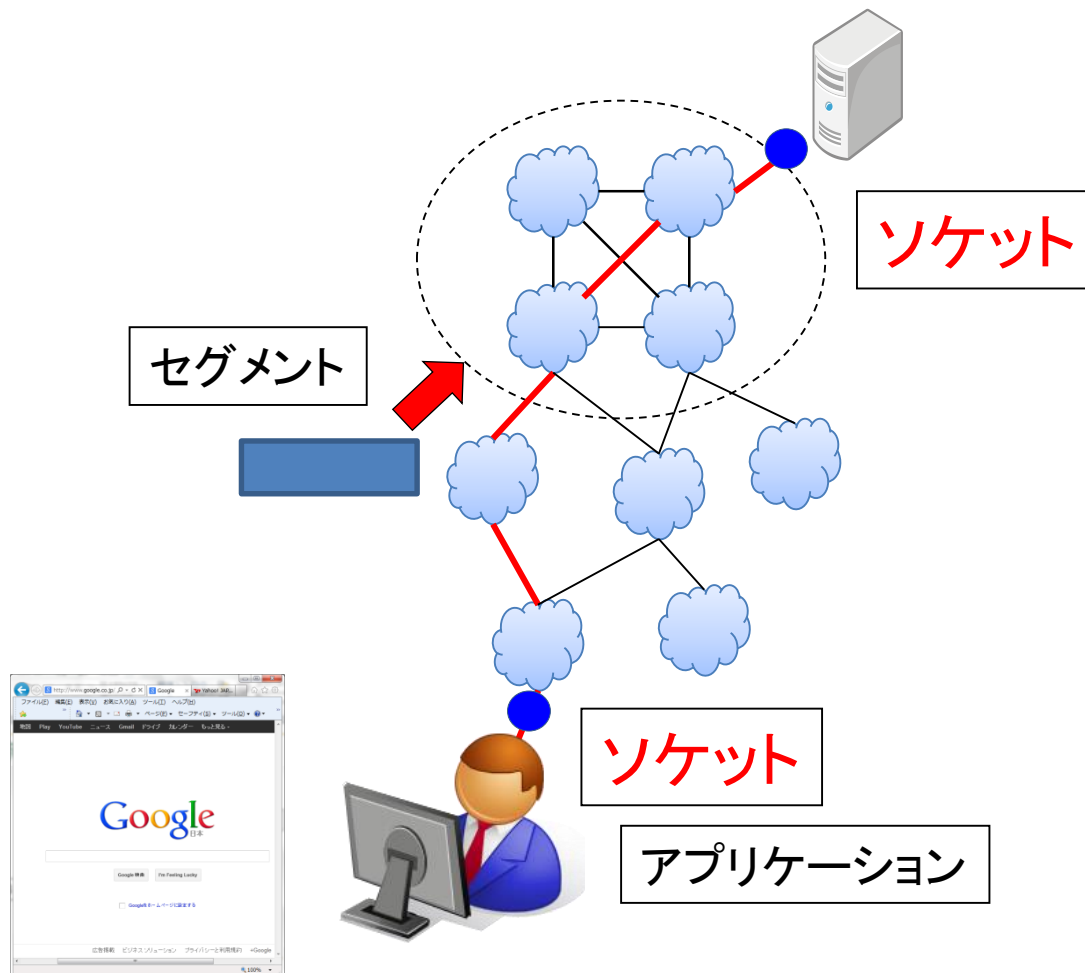
トランスポート層プロトコル(5.5節)

プロトコルの階層化: 送受信データの流れ



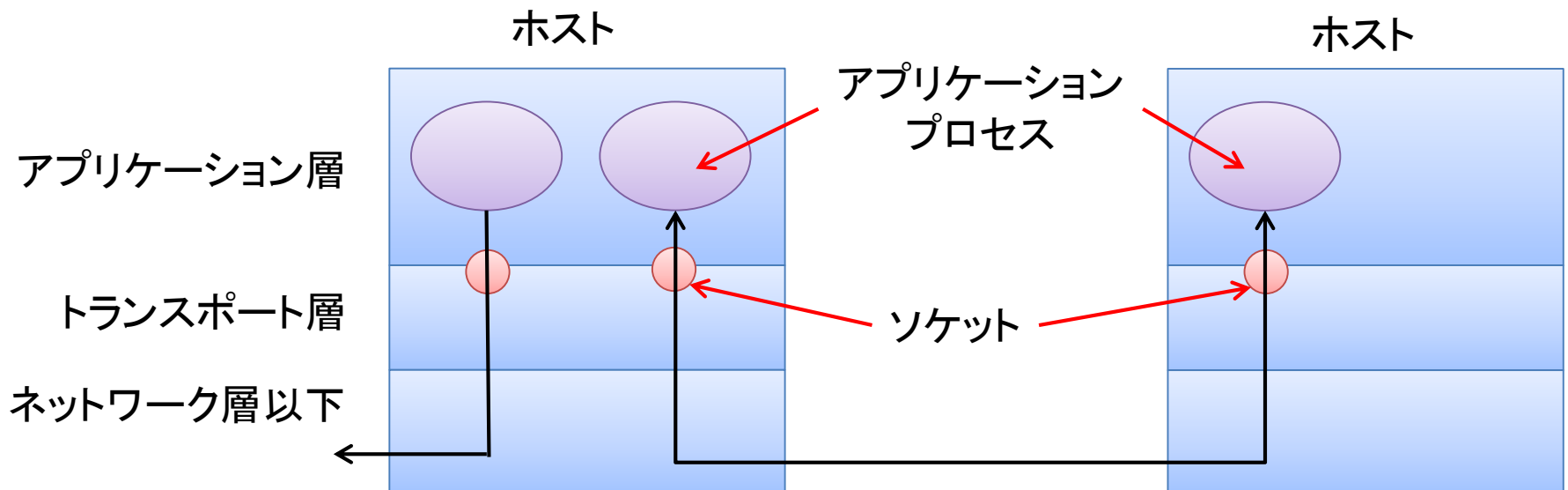
トランスポート層の役割

- アプリケーションプロセス間での通信機能を提供

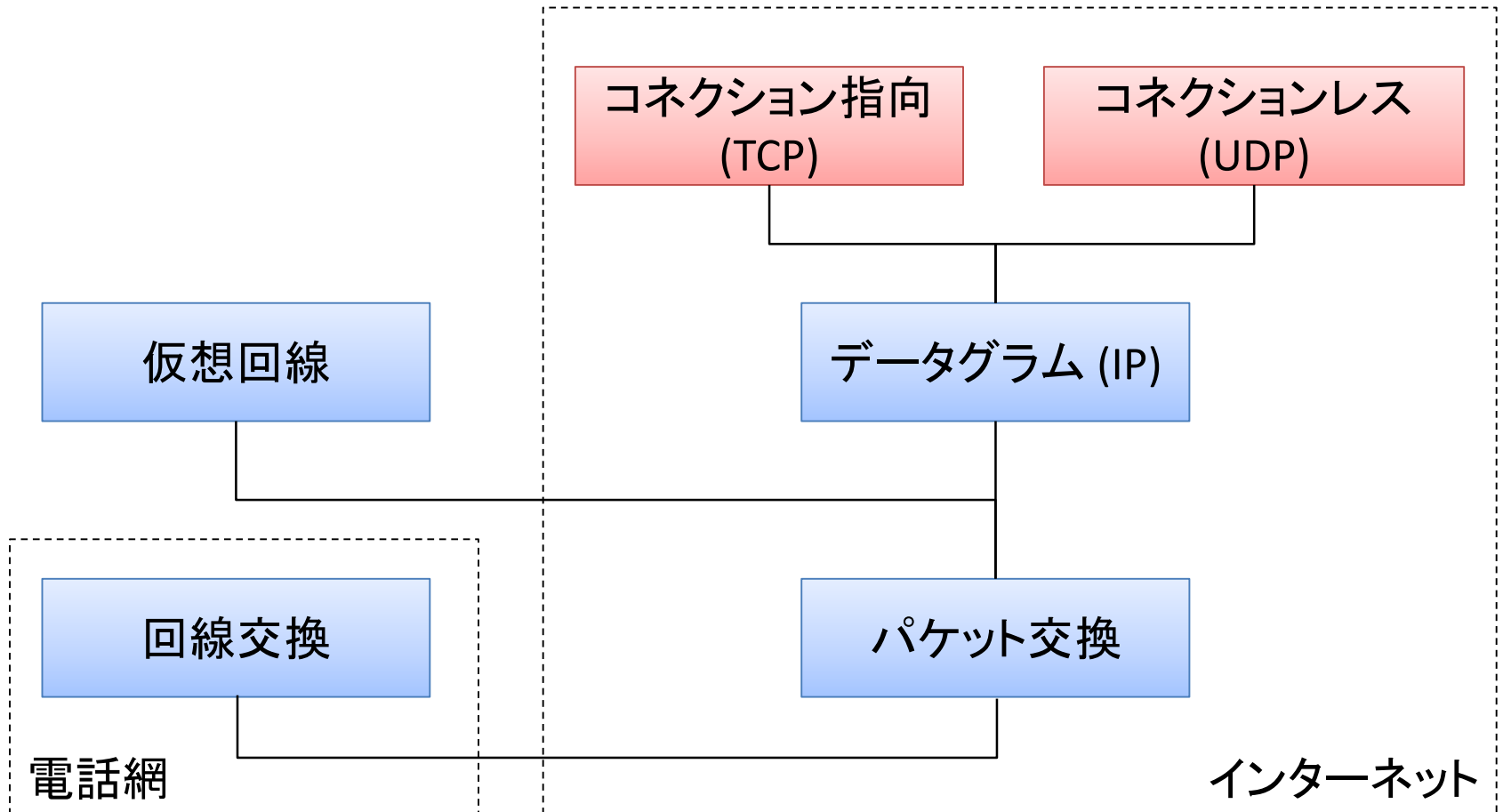


トランスポート層の役割

- アプリケーション間での通信機能を提供
- **ソケット**: トランスポート層がアプリケーション層に提供する通信サービスの端点
 - ソケットで相手のアプリケーションプロセスを指定
- ソケットは, IPアドレス + **ポート番号** で表現される



通信方式の分類 (再掲)



パケット交換（ネットワーク層）の問題

送りたい情報（メッセージ）

1. パケットに分割

パケット1 パケット2 ... パケットn

2. 様々な送信処理

処理1
処理2
処理3
処理4
....



エンドホストA
(コンピュータ)



ルータ
(中継装置)



エンドホストB
(コンピュータ)

パケットn ... パケット2 パケット1



3. パケットごとに送信・伝送

パケット交換（ネットワーク層）の問題

受信した情報（メッセージ）

6. パケットを集めて組み立て

パケット1

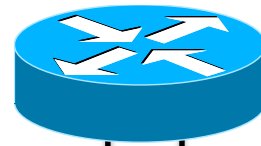
パケット2

...

パケットn



エンドホストA
(コンピュータ)



ルータ
(中継装置)



エンドホストB
(コンピュータ)

5. 再送
などの処理

パケットn

...

パケット2

パケット1

1はOK!

2を再送!

4. 応答 (ACK)

...

コネクション指向通信とコネクションレス通信

- コネクション指向通信

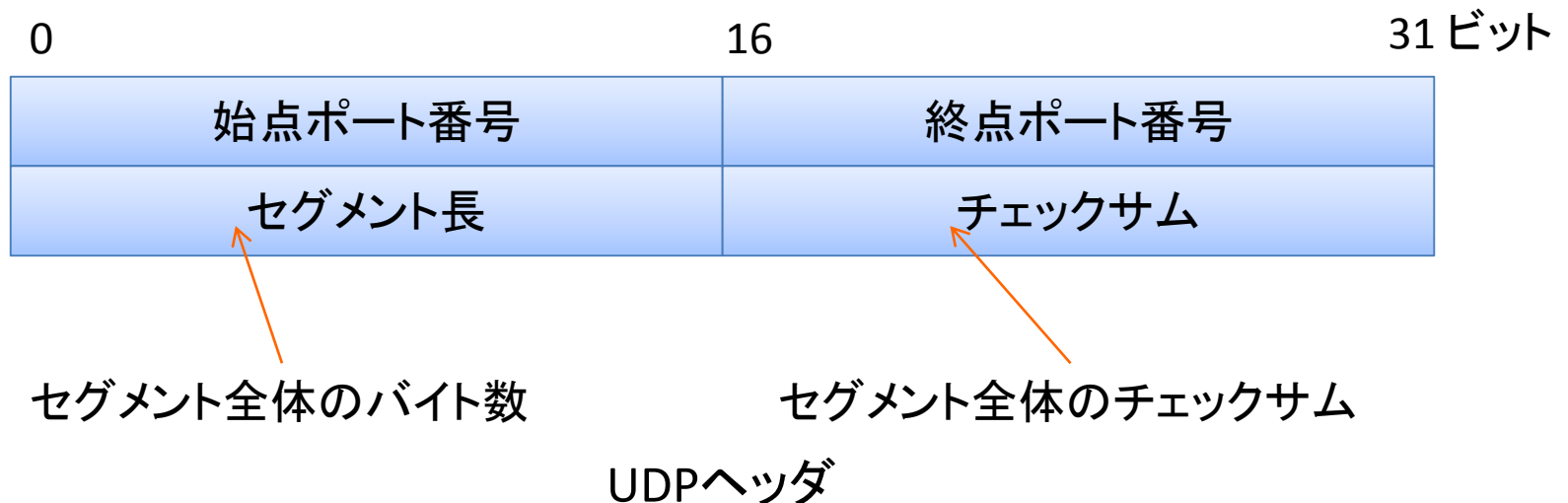
- 通信前にエンドホスト間にコネクションを確立, 終了時に解放
 - エンドホストのみが状態を保持
- 信頼性を保証した通信が可能
- TCP (Transmission Control Protocol, 後述)

- コネクションレス通信

- コネクションの確立・解放は不要
- 信頼性の保証なし
- UDP (User Datagram Protocol, 後述)

UDP

- コネクションレス型のトランスポート層プロトコル
- ポート番号による多重化・逆多重化
 - 異なるアプリの通信を, ポート番号を変えることで, IPの通信にまとめている = 多重化



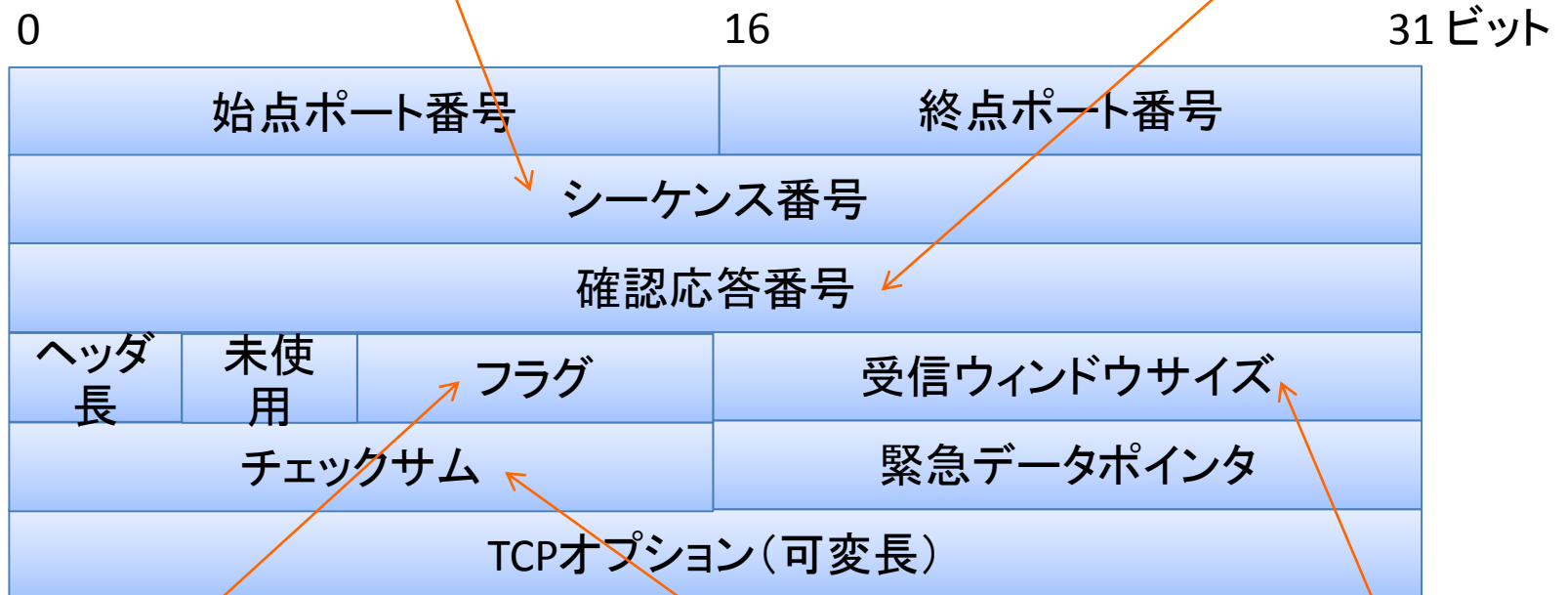
TCP

- コネクション型のトランスポート層プロトコル
 - 通信前にコネクションの確立, 通信終了後に解放
- ポート番号による多重化・逆多重化 (UDPと同様)
- 信頼性保証
 - シーケンス番号 → 順序どおり & 欠落なしの伝送を保証
 - チェックサム → データ誤りを検出
 - 確認応答セグメント (Ack segment) → 送信側へフィードバック
 - データセグメントに相手から受信したデータへのAckを付加
- 流量制御: 受信ホストの能力に合わせて送信速度を調整
- 輻輳制御: ネットワークが過度に混在しないように送信速度を調整
 - 輻輳 (ふくそう) = ネットワーク中での通信の渋滞

TCPヘッダ

- ・シーケンス番号はデータ1バイトごとに割り当てられる
- ・ペイロードの先頭データのシーケンス番号を示す

次に受信したいデータの
シーケンス番号を示す



コネクション管理に使用

セグメント全体の
チェックサム

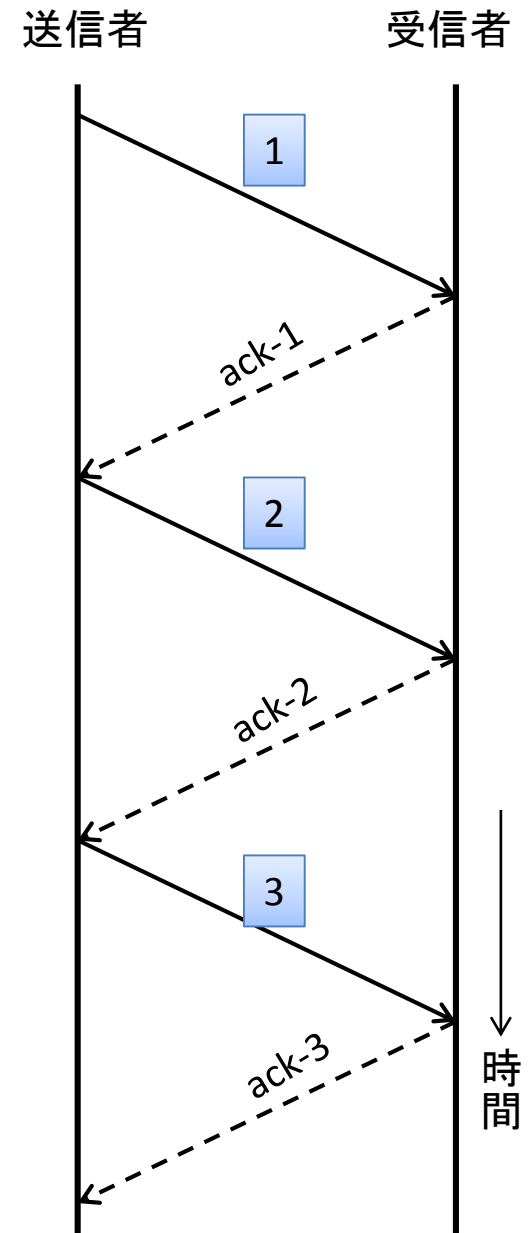
受信側のバッファの
空き

TCPの送信制御

- Stop & Waitプロトコル
 - “データセグメントを1つ送信 + Ackセグメント受信を待つ”の繰り返し
 - 効率が悪い
- スライディングウィンドウプロトコル (TCPはこちら)
 - Ackを未受信でも複数のデータセグメントを送信
 - ウィンドウ: Ack未受信で送信可能なシーケンス番号の範囲
 - Ack受信ごとに“ウィンドウ”がずれ, 新たなデータセグメント送信
→ “スライディングウィンドウ”

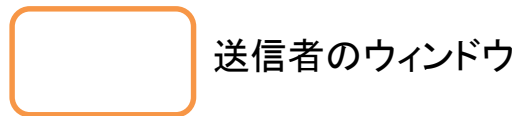
Stop & Wait プロトコル

- データセグメントを1つ送信
+ Ackセグメント受信を待つ
- エンドホスト間の通信遅延(片道)
が50msec とすると, Ack受信まで
に100msec
→ 1秒間に10セグメントしか送ることが
できない



(a) ストップ・アンド・ウェイト

スライディング ウィンドウプロトコル

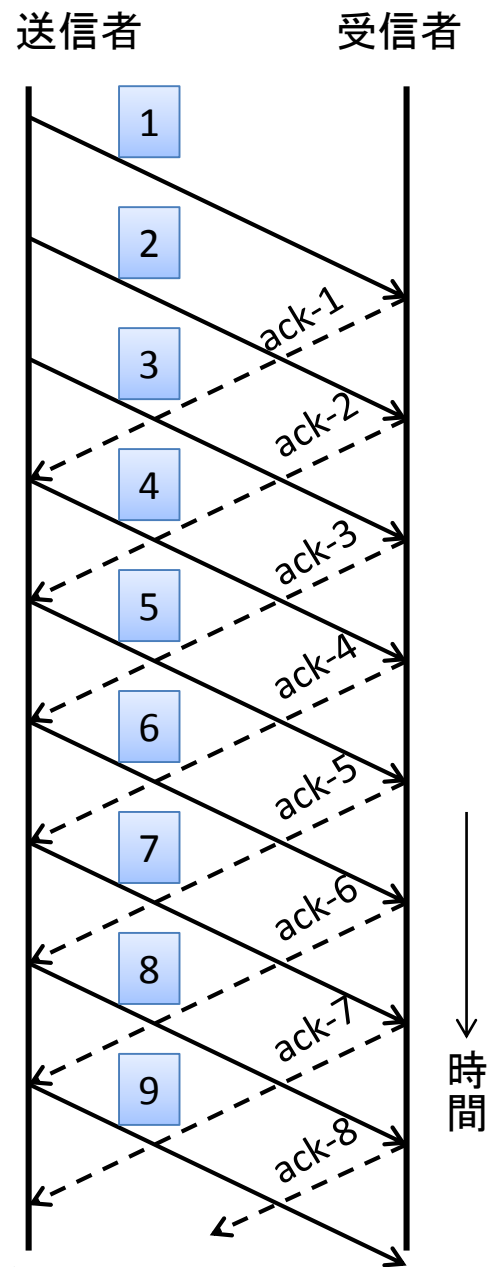
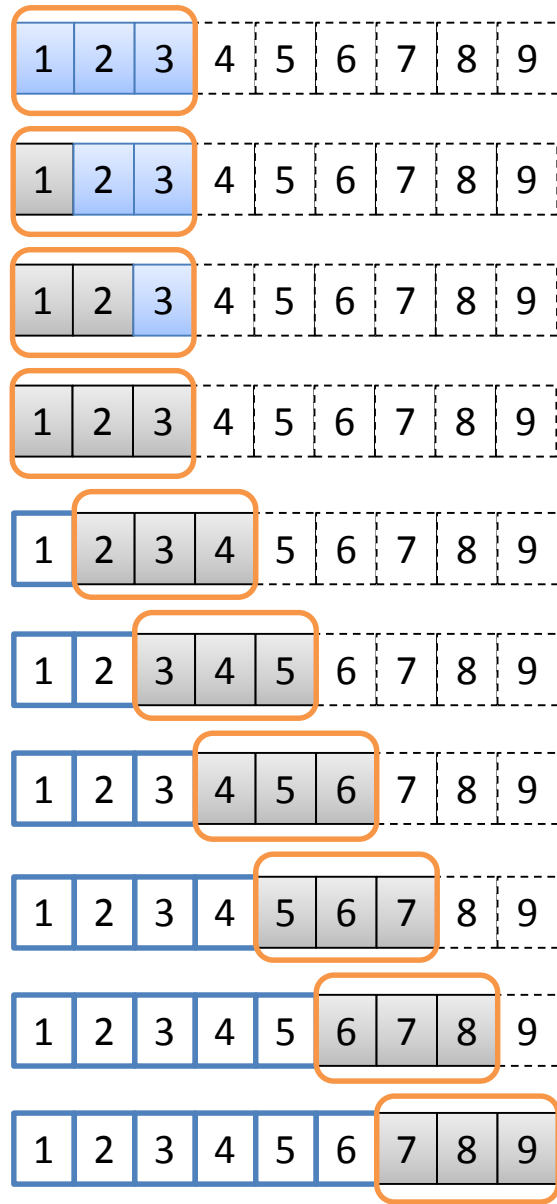


x 送信可能なセグメント

× 送信済かつACK未受信のセグメント

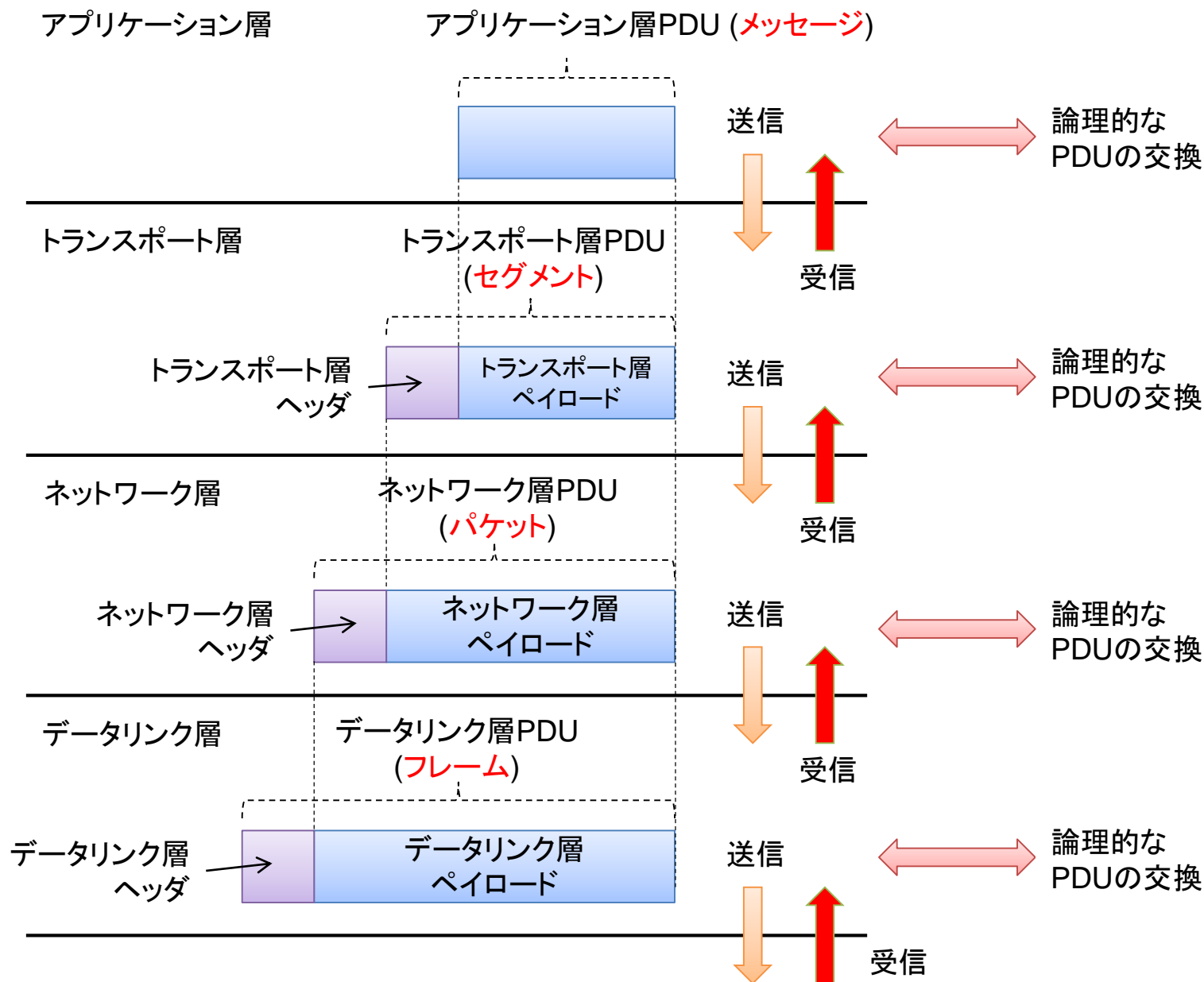
x ACK受信済のセグメント

x 送信不可のセグメント



(b) スライディングウィンドウ

プロトコルの階層化: 送受信データの流れ(まとめ)



本日のまとめ

- インターネット1: ネットワークの基礎(5章)
 - インターネットの構造(5.1節)
 - プロトコルの階層化(5.2節)
 - 物理層とデータリンク層プロトコル(5.3節)
 - ネットワーク層プロトコル(5.4節)
 - トランスポート層プロトコル(5.5節)
- 来週は6章を読んで来て下さい
- 第三回課題も忘れずに(6/15締め切りです)