データ収集システムの技術

長坂康史
nagasaka@cc.it-hiroshima.ac.jp
広島工業大学
情報学部情報工学科

データ収集システムの技術

- 達成目標:
  - データ収集システム技術、特に、システムで多く利用されるネットワーク技術を理解し、その重要性を知る。また、ネットワークを利用した データ収集システムの利点欠点を理解し、その利用ができるようになる。
- 内容:
  - データ収集システムに応用されるネットワーク技術の基礎を説明する。 また、システムの一部としてのネットワーク技術の重要性も理解させる。

目次

- データ収集システム技術
  - オブジェクト指向技術
  - システム設計技術 – UML
  - ネットワーク技術

Object-Oriented Technology

オブジェクト指向技術

構造化プログラミング

- プログラム（情報システム）
  - ある特定の処理をさせるための命令の集まり
    - プロシージャ（手続き）
  - 大規模プログラム（情報システム）では
    - プロシージャの数 → 大
    - プログラマの数 → 大
    - プログラム開発期間 → 大

    - プログラムの分割化
    - 分割された部分の開発
    - 統合

オブジェクト指向プログラミング

- プログラムに求められるもの
  - 柔軟性
  - 保守の容易性
  - 高信頼性
  - 情報(データと処理方法)の保存の統一化
    - データと処理

構造化プログラミング
　データとその処理方法は別々に保存

- オブジェクト指向プログラミング
  - 世の中のさまざまな実体をオブジェクトととらえて、実世界ではこのオブジェクト同士がメッセージをやり取りすることで成り立っていると考え、それをコンピュータ上で模倣(シミュレート)する技法

オブジェクト指向技術(１)

- オブジェクト指向技術
  - Object-Oriented Technology
  - 世の中のさまざまな実体をオブジェクトととらえて、コンピュータでシミュレートする技術

- 三つの重要なキーワード
  - オブジェクト　Object
  - クラス　Class
  - メッセージ　Message

オブジェクト指向技術(２)

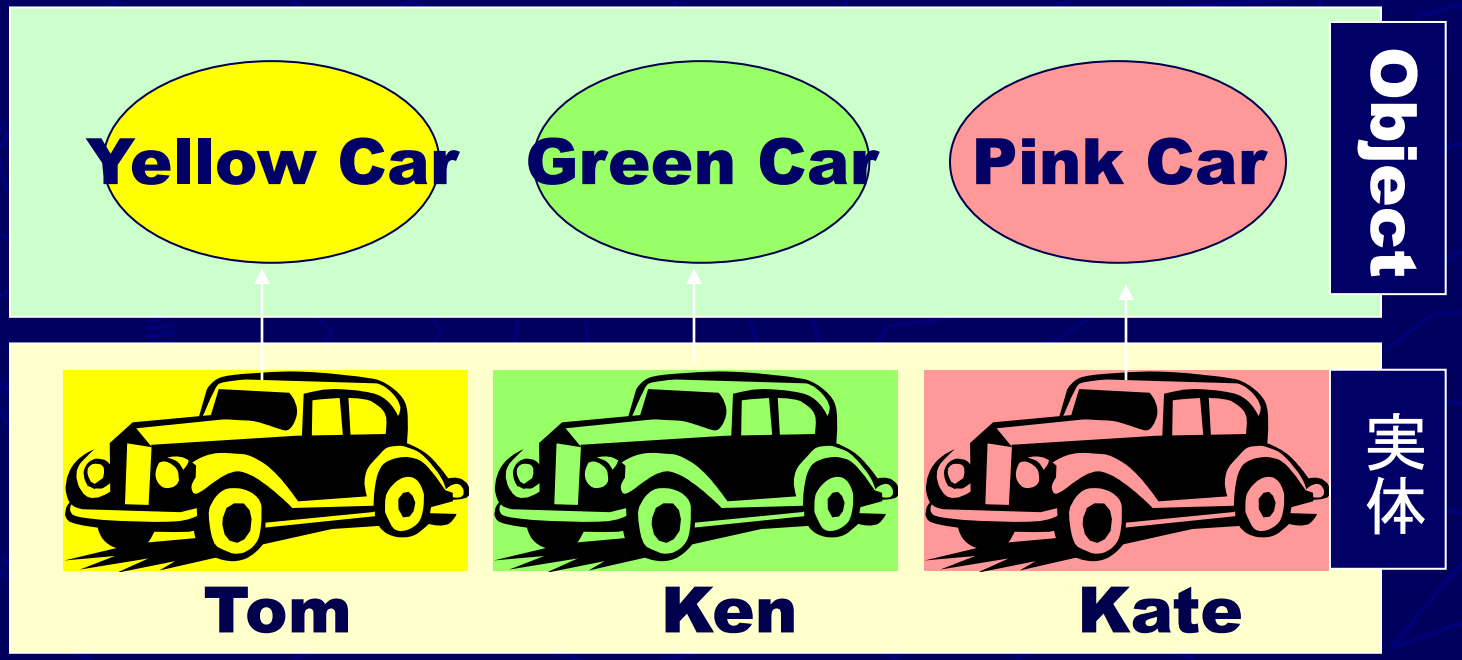
- そのほかの基本用語
  - 継承（Inheritance）
  - メソッド（Method）
  - インスタンス（Instance）
  - カプセル化（Encapsulation）
  - 抽象化（Abstraction）
  - 多相性(多態性、多様性）（Polymorphism）
  - オーバロードィング（Overloading）
  - オーバライディング（Overriding）

オブジェクト（１）

- オブジェクト（Object）
  - 実世界にある「もの」のある「状態」を表すもの
    - データとその処理方法（操作）が一体
  - 属性（Attribute）
    - オブジェクトの持つ変数（パラメータ：Parameter）
    - データは外部からは隠蔽（Data Hiding）
    - → カプセル化（Encapsulation）
  - 操作（Operation）
    - 隠蔽されたデータを操作する手段を提供
    - 操作を実装したものがメソッド（Method）

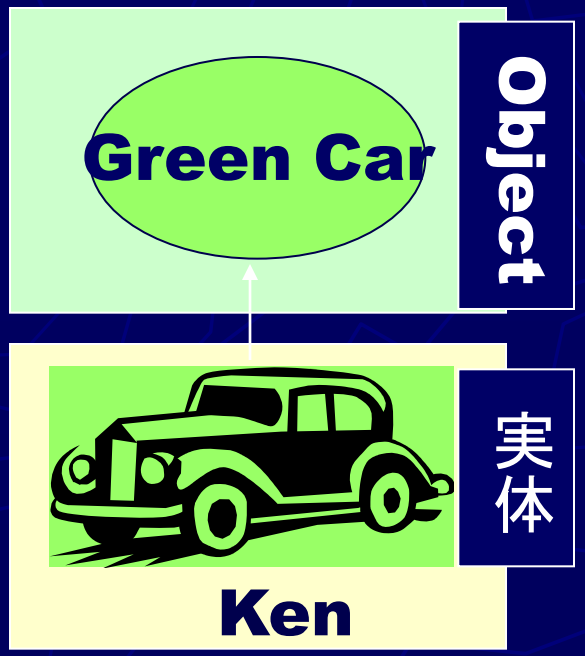
オブジェクト(２)

- オブジェクトの例
  - Tomの車： 黄色、停止中, ガソリン8割, ...
  - Kenの車： 緑色、時速50キロで走行中, ガソリン9割, ...
  - Kateの車： 桃色, 後退中, ガソリン5割, ...



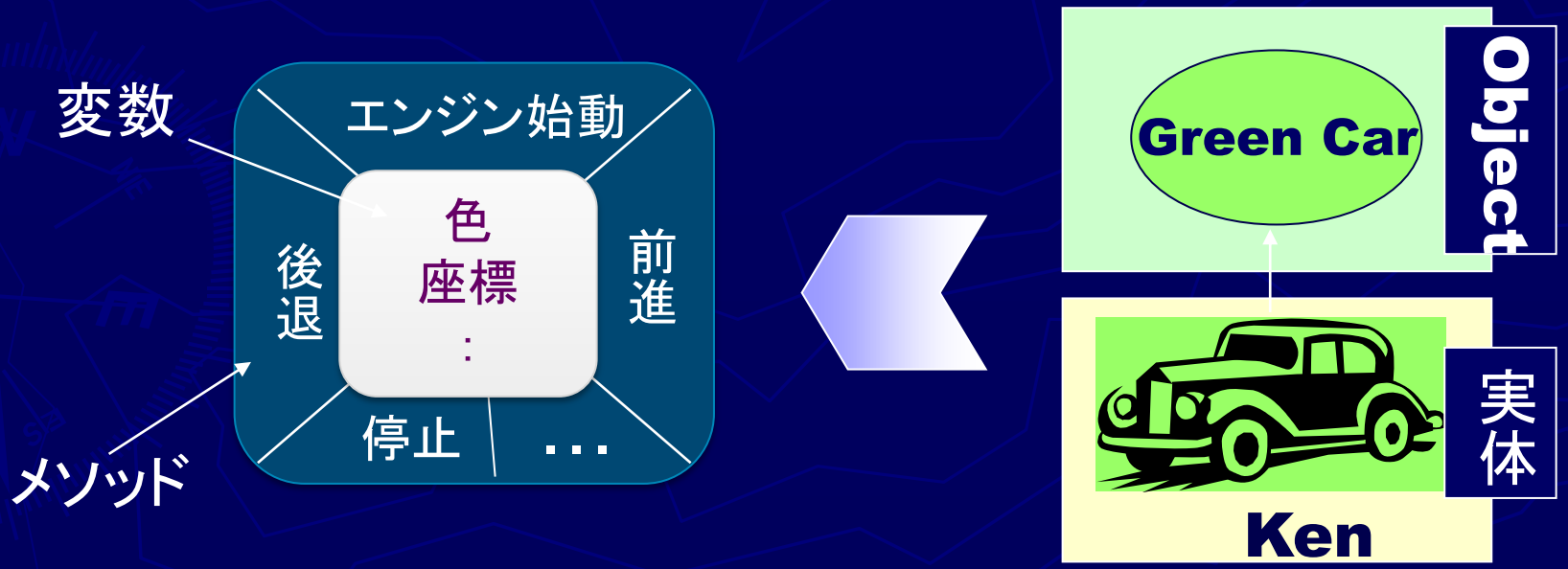
オブジェクト(３)

- オブジェクトの例
  - Ken の車の属性
    - 色： 緑
    - タイヤ：４
    - ドアの数： ５
    - ：
    - 座標 (x, y, z)
    - 加速度
  - Ken の車の操作
    - エンジン始動
    - 移動速度の加速、減速
    - 移動の停止



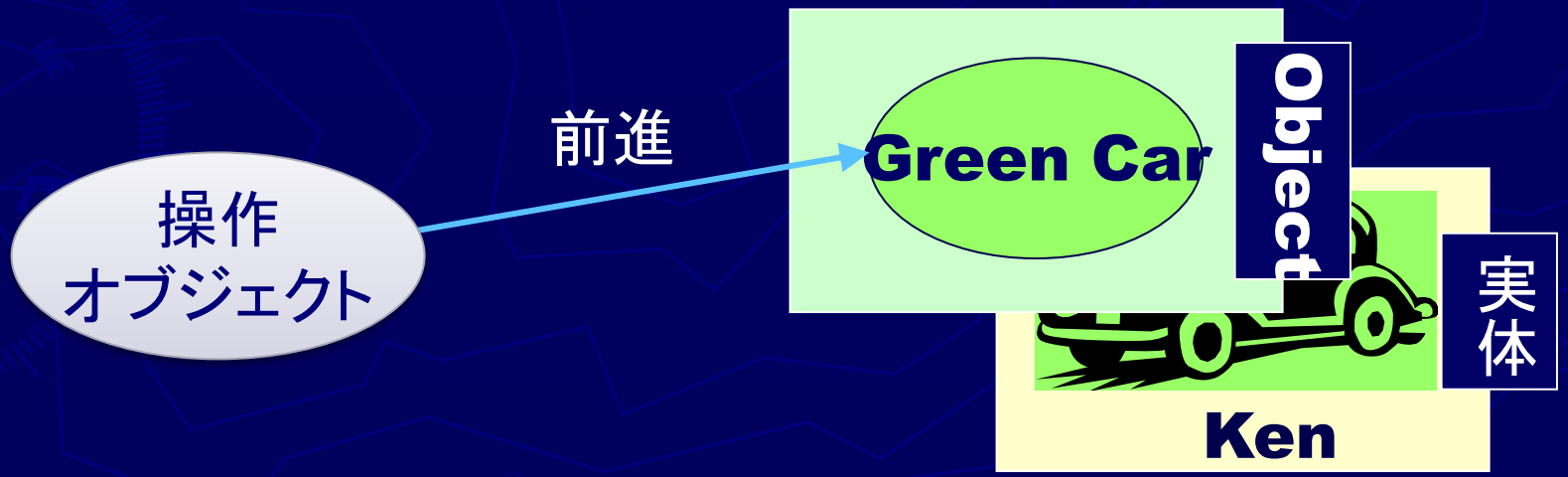
オブジェクト(４)

- オブジェクトのイメージ
  - Ken の車の属性
    - ⇒ オブジェクト内部に隠蔽されている
  - Ken の車の操作
    - ⇒ 変数の周りに位置し、メソッドを介してでないと変数は変更できない



メッセージ

- メッセージ（Message）
  - オブジェクト間でのやりとりされるもの
  - あるオブジェクトに対して、何らかの操作をするためにそのオブジェクトのメソッド（操作）に送られるもの
  - メッセージの送り手と受け手が存在



クラス(１)

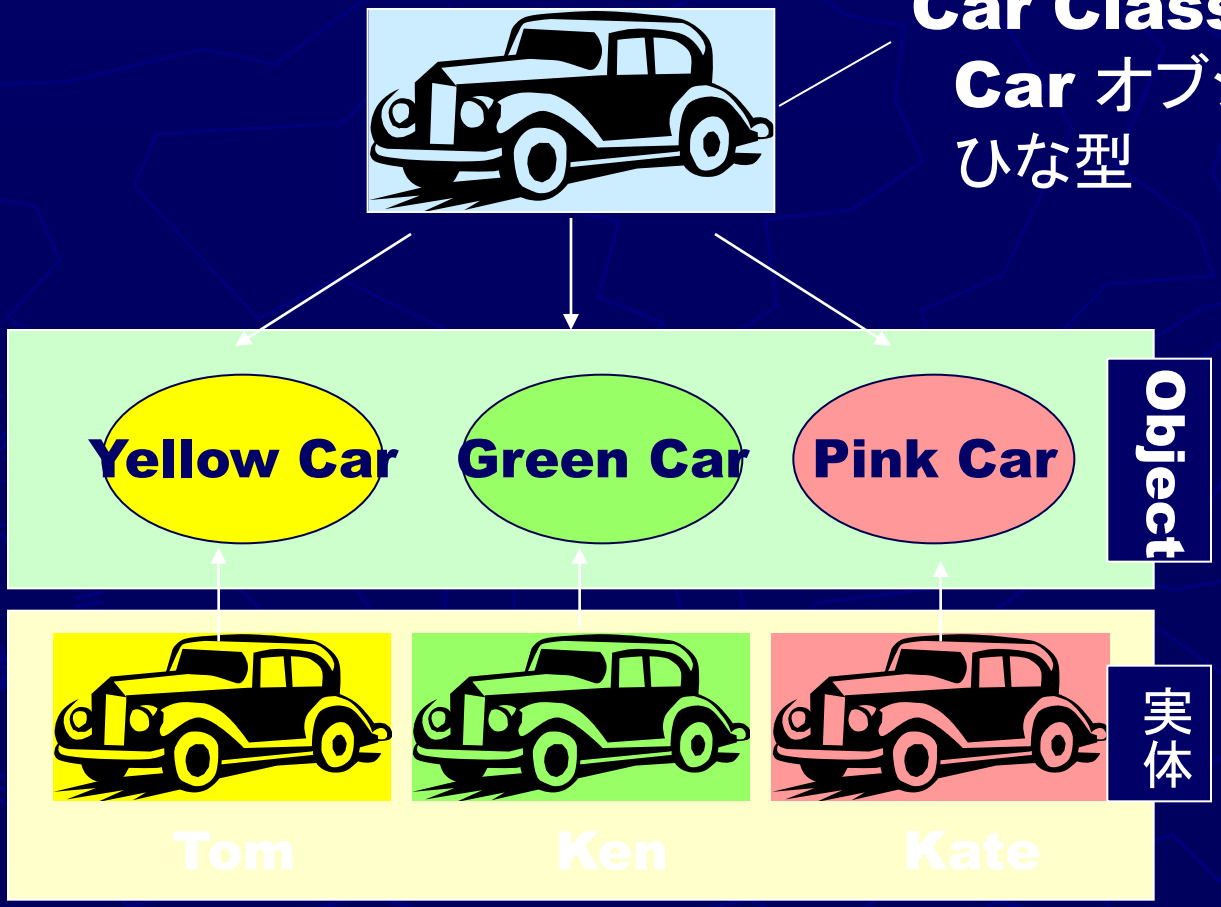
- クラス(Class)
  - オブジェクトのひな型(原型)
    - オブジェクトは変数に値が入力された実体
    - クラスは変数の定義はされているが、値は未入力
    - クラスだけでは実体は存在しない
  - オブジェクトはクラスから生成される
    - 生成時に、変数に値が代入される

クラス(２)

- クラスとオブジェクト
  - クラスを具現化(Instantiation)したものがインスタンス(Instance)

クラス(３)

- クラスとオブジェクト(２)

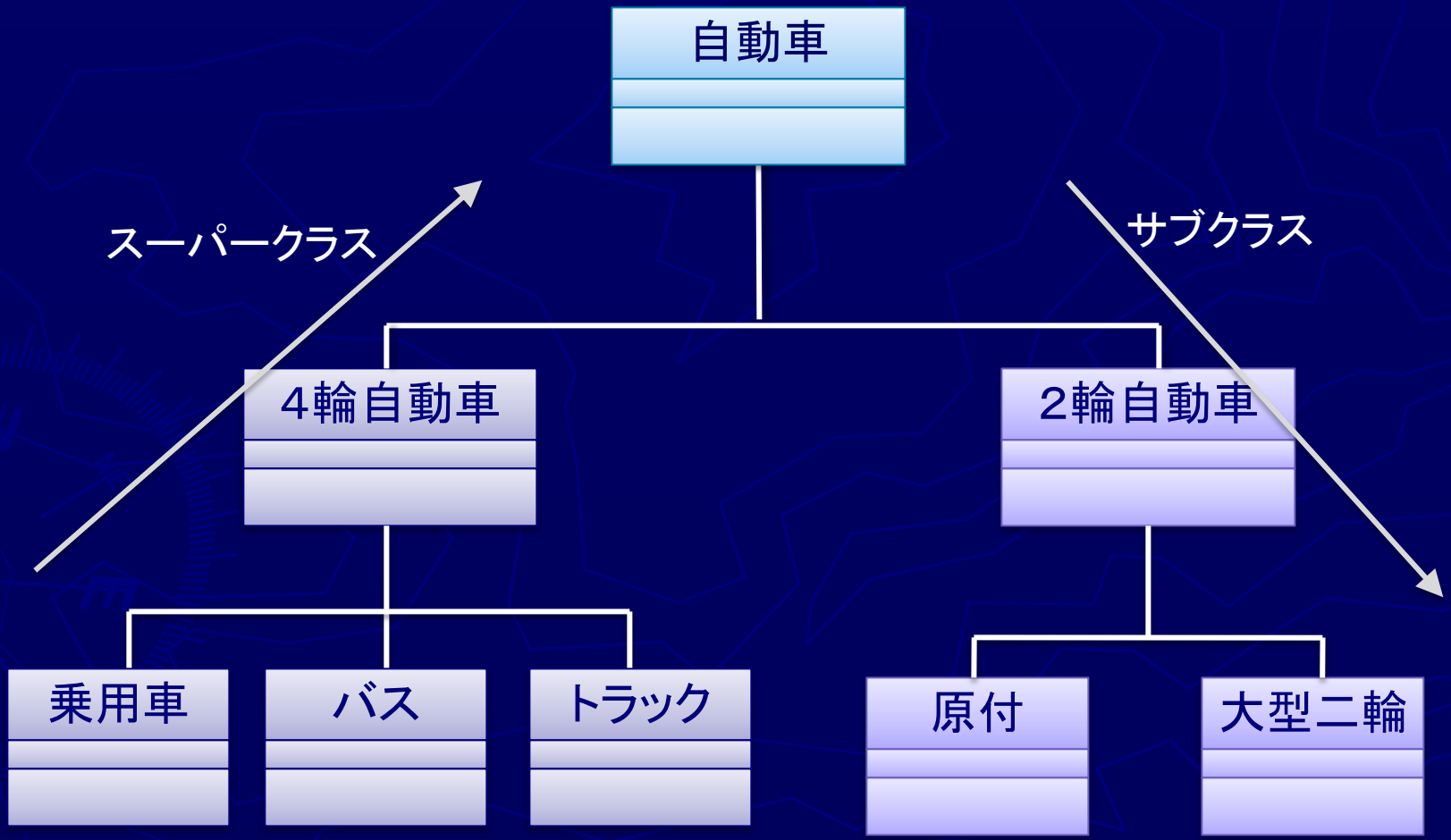


クラス(４)

- 継承(Inheritance)
  - あるクラスの特性をそのまま受け継ぐこと
  - スーパークラス(Superclass)
    - 継承されたクラス
    - 上位層のクラスと考える
  - サブクラス(Subclass)
    - 継承したクラス
    - 下位層のクラスと考える

クラス(５)

- スーパークラスとサブクラス



クラス(６)

- 継承
  - 単一継承(Single Inheritance)
    - クラスが必ず一つのスーパークラスしか持たないような継承方式。
  - 多重継承(Multiple Inheritance)
    - クラスが複数のスーパークラスを持った継承方式。

まとめ：オブジェクト指向技術

- オブジェクト指向技術
  - Object-Oriented Technology
  - 世の中のさまざまな実体をオブジェクトととらえて、コンピュータでシミュレートする技術

- 三つの重要なキーワード
  - オブジェクト　Object
  - クラス　Class
  - メッセージ　Message

UML: Unified Modeling Language

UML：統一モデリング言語

UML概要

- UML（Unified Modeling Language）
  - 統一モデリング言語
  - オブジェクト指向モデルを記述する言語
    - オブジェクト指向分析設計時に考えを具象化する道具
    - 技術者間のコミュニケーションの道具

- UMLの歴史
  - 1994 OMTの第二版（J.Rumbaugh + G.Booch）
  - 1995 UMLの初期版（J.R+G.B+I.Jacobson）
  - 1997 UML 1.0
  - 1997 UML 1.1 OMGで承認 標準化
    - OMG: Object Management Group
  - 1999 UML 1.3
  - 2001 UML 1.4
  - 2003 UML 1.5
  - 2005 UML 2.0

UMLの仕組み

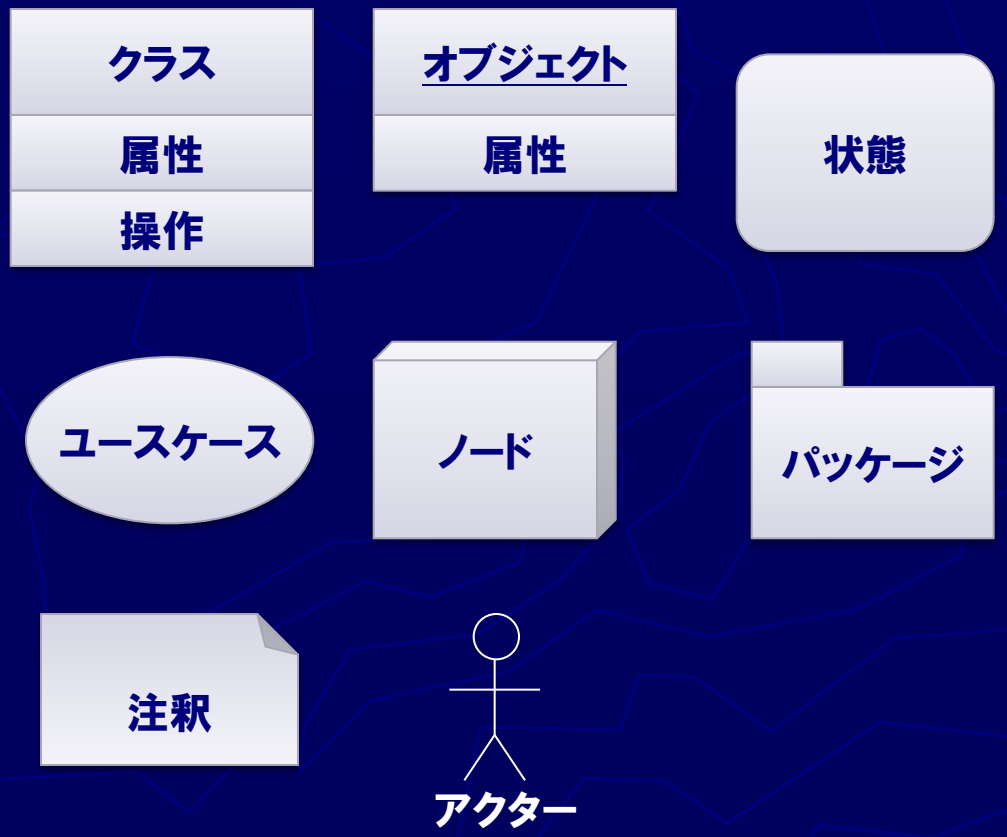
- ビュー（View）
  - システムをモデル化してできるもの
    - さまざまな側面から表現するため複数個存在
  - 複数のダイアグラムから構成
- ダイアグラム（Diagram）
  - ビューの内容を表すグラフ
- モデル要素（Model Element）
  - ダイアグラムで利用される概念
    - UMLではクラス、オブジェクト、メッセージ、モデル要素間の関係など
- その他
  - コメントや情報の挿入などの一般的な機能
  - 拡張への仕組み

UML2.0 Diagram

- ユースケース図 (Use Case Diagram)
- 構造図 (Structural Diagrams)
  - 論理的な図
    - クラス図 (Class Diagram)
    - オブジェクト図 (Object Diagram)
    - パッケージ図 (Package Diagram)
  - 物理的な図
    - コンポジット構造図 (Composite Structure Diagram)
    - コンポーネント図 (Component Diagram)
    - 配置図 (Deployment Diagram)
- 振る舞い図 (Behavioral Diagrams)
  - ステートマシン図 (State Machine Diagram)
  - アクティビティ図 (Activity Diagram)
  - 相互作用図 (Interaction Diagrams)
    - シーケンス図 (Sequence Diagram)
    - コミュニケーション図 (Communication Diagram)
    - 相互作用概要図 (Interaction Overview Diagram)
    - タイミング図 (Timing Diagram)

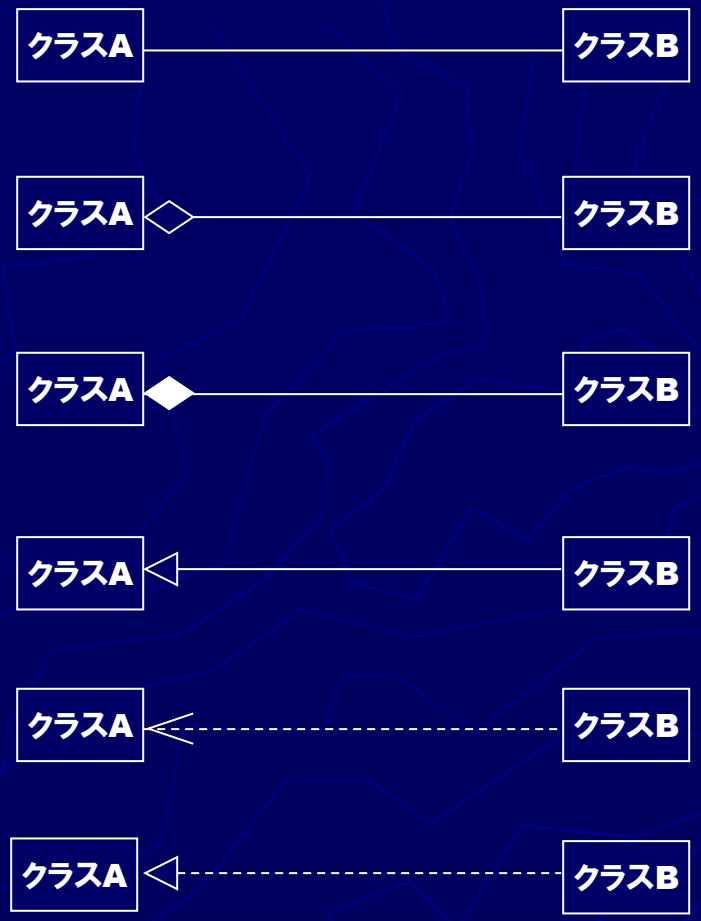
モデル要素(１)

- 図形シンボル
  - クラス
  - オブジェクト
  - 状態
  - ユースケース
  - ノード
  - パッケージ
  - 注釈
  - アクター
  - ・・・



モデル要素(２)

- 関係シンボル
  - 関連（Association）
  - 集約（Aggregation）
  - コンポジション（Composition）
  - 汎化（Generalization）
  - 依存（Dependency）
  - 実現（Realization）



クラス図

- クラス図（Class Diagram）
  - クラスとそれぞれのクラスの関係を表記した図
    - 分類子（Classifier）
    - 関係（Relationship）
  - 分類子（Classifier）
    - クラス（Class）
    - ステレオタイプ（Stereotype）
  - 関係（Relationship）
    - 関連（Association）
    - 集約（Aggregation）
    - コンポジション（Composition）
    - 汎化（Generalization）
    - 依存（Dependency）
    - 実現（Realization）

クラスの表記

- クラスの表記
  - 属性と操作の表記

  - 三つの区画
    - 1: 名前区画　ークラスの名前・プロパティ
      - クラス名　｛　プロパティ　｝
    - 2: リスト区画（１）　ークラスの持つ属性
      - 可視性　属性名：型　＝　初期値
    - 3: リスト区画（２）　ークラスの持つ操作
      - 可視性　方向　操作名（引数：型、・・・）：戻り値の型

クラスの属性

- 属性（Attribute）
  - オブジェクトの持つフィールド（データ）の定義
    - メンバ変数、インスタンス変数、データメンバ

(1) 可視性　属性名：型　＝　初期値
- - tire　：int　＝　4

(2) 可視性　属性名：型［多重度］　＝　初期値｛プロパティ｝
- + name : string [1..2]　＝　“Tanaka” { frozen }

クラスの操作

- 操作（Operation）
  - オブジェクトに送られるメッセージによって起動する動作
    - メンバ関数、メソッド

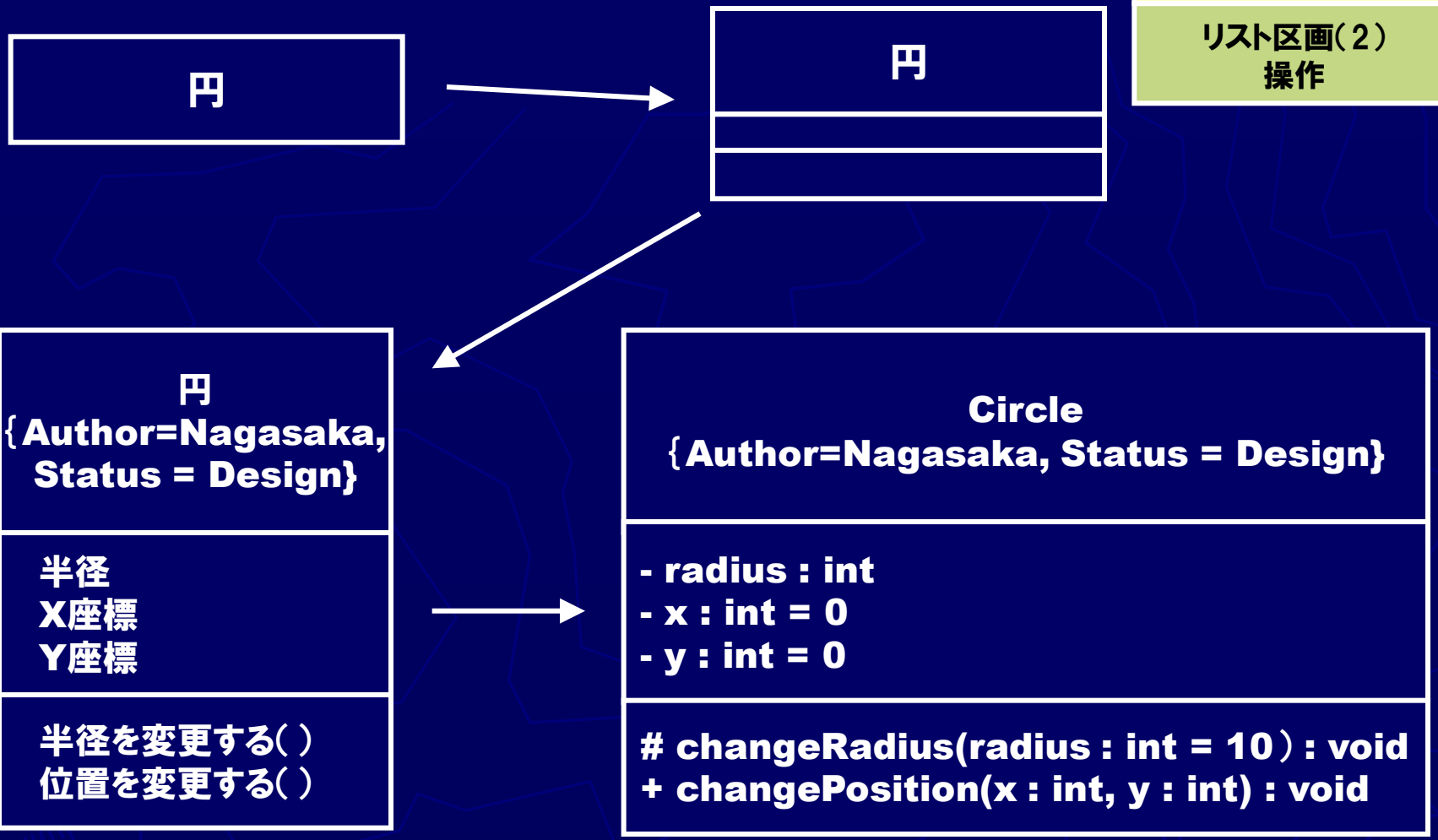
(1) 可視性　操作名（引数名：引数の型）:戻り値の型
- ＋　set (speed : int　) : void

(2) ステレオタイプ　可視性 操作名（入出力種別　引数名：型=デフォルト値）:戻り値の型{プロパティ}
- <<accessor>> + setSpeed (in speed : int = 0) : void

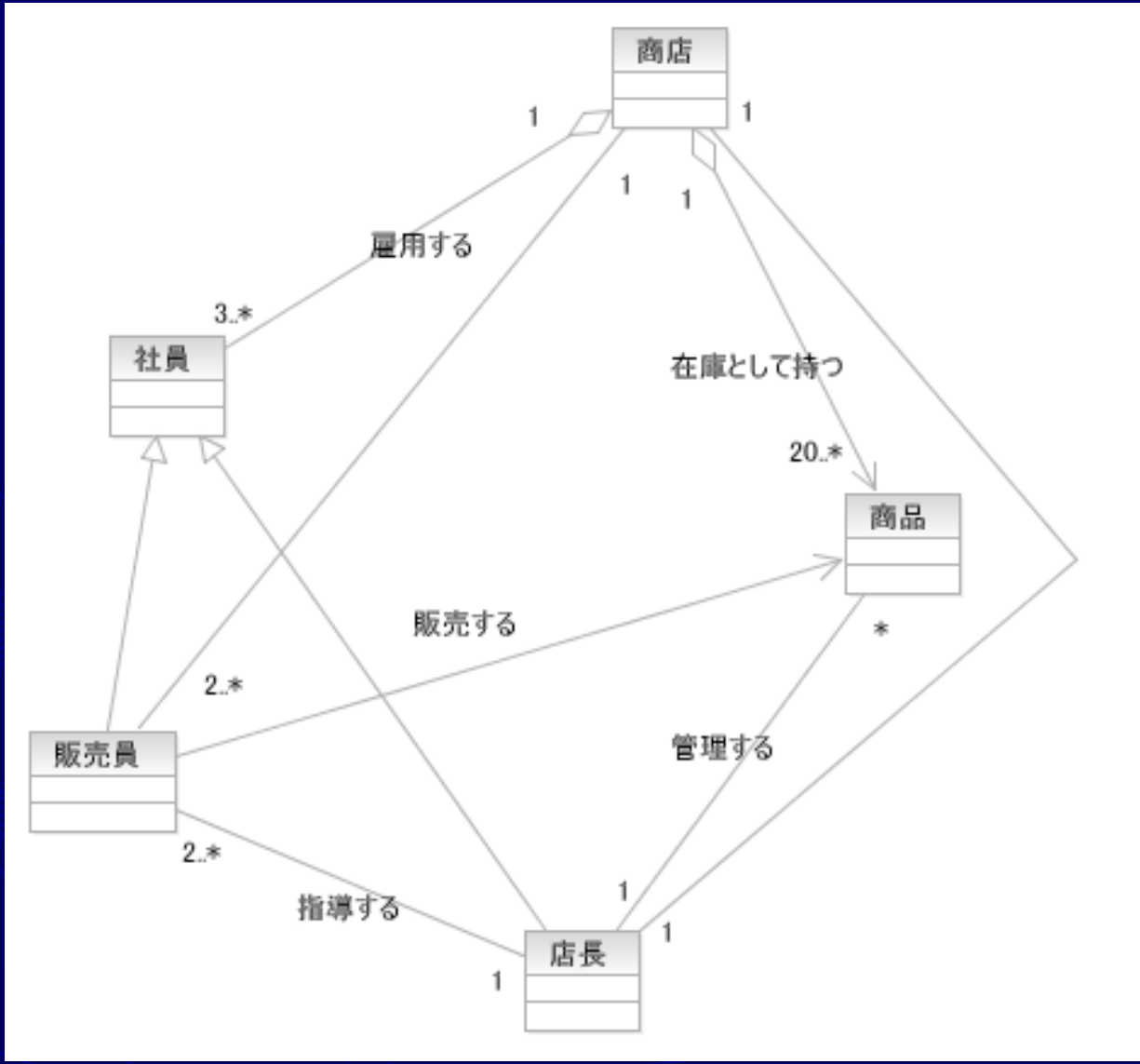
可視性

- 可視性（アクセス制御）
  - Public「+」
    - どのクラスからでも参照可能
  - Protected「#」
    - 同一クラス内、および、派生クラスからのみ参照可能
  - Private「-」
    - 同一クラス内からのみ参照可能
  - Package「～」
    - 同一パッケージ内のクラスからのみ参照可能

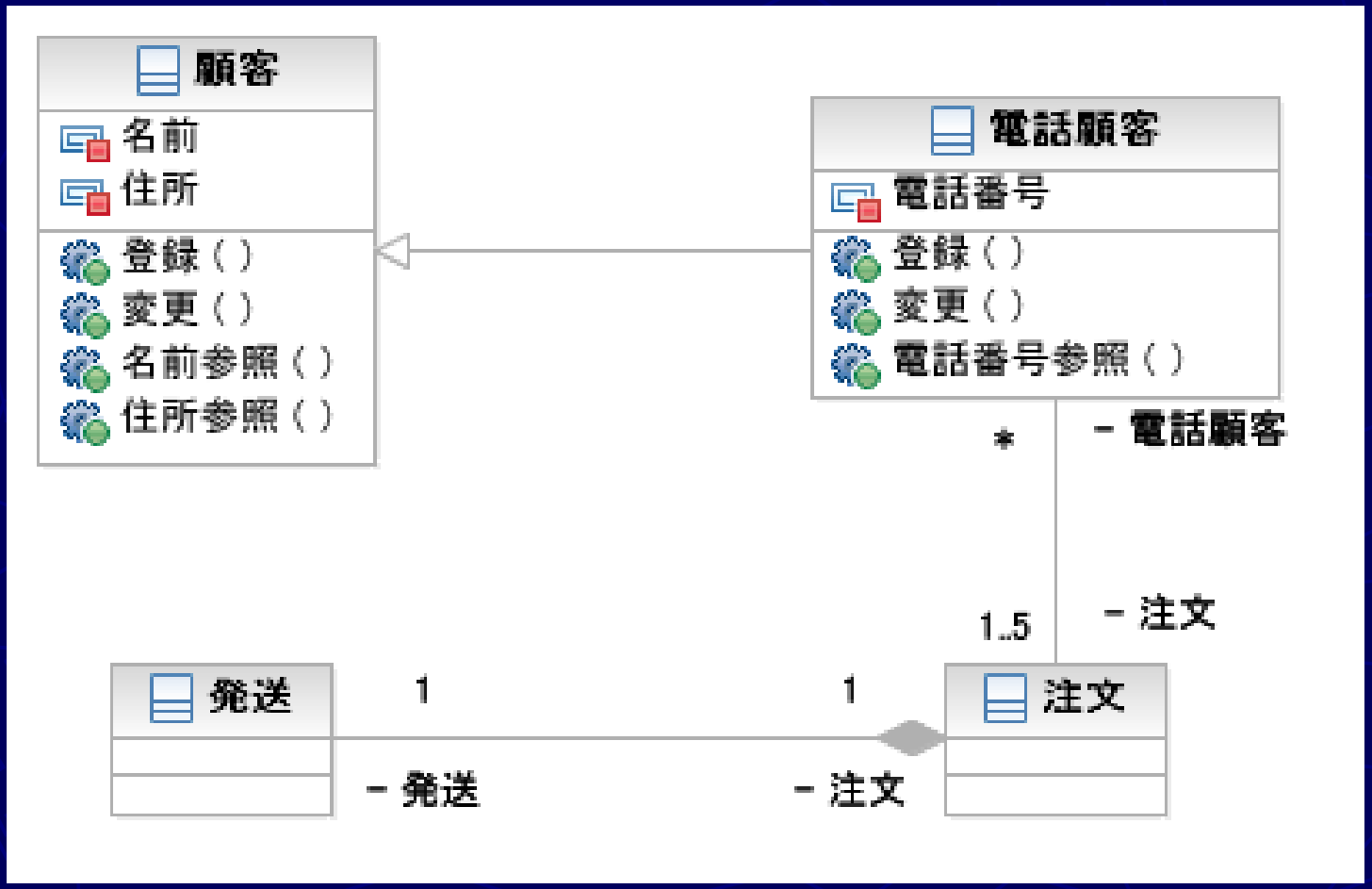
クラス表記の例


名前区画
名前・プロパティ
リスト区画（１）
属性
リスト区画（２）
操作
円
円
円
{Author=Nagasaka,
Status = Design}
半径
X座標
Y座標
半径を変更する( )
位置を変更する( )
Circle
{Author=Nagasaka, Status = Design}
- radius : int
- x : int = 0
- y : int = 0
changeRadius(radius : int = 10) : void
+ changePosition(x : int, y : int) : void

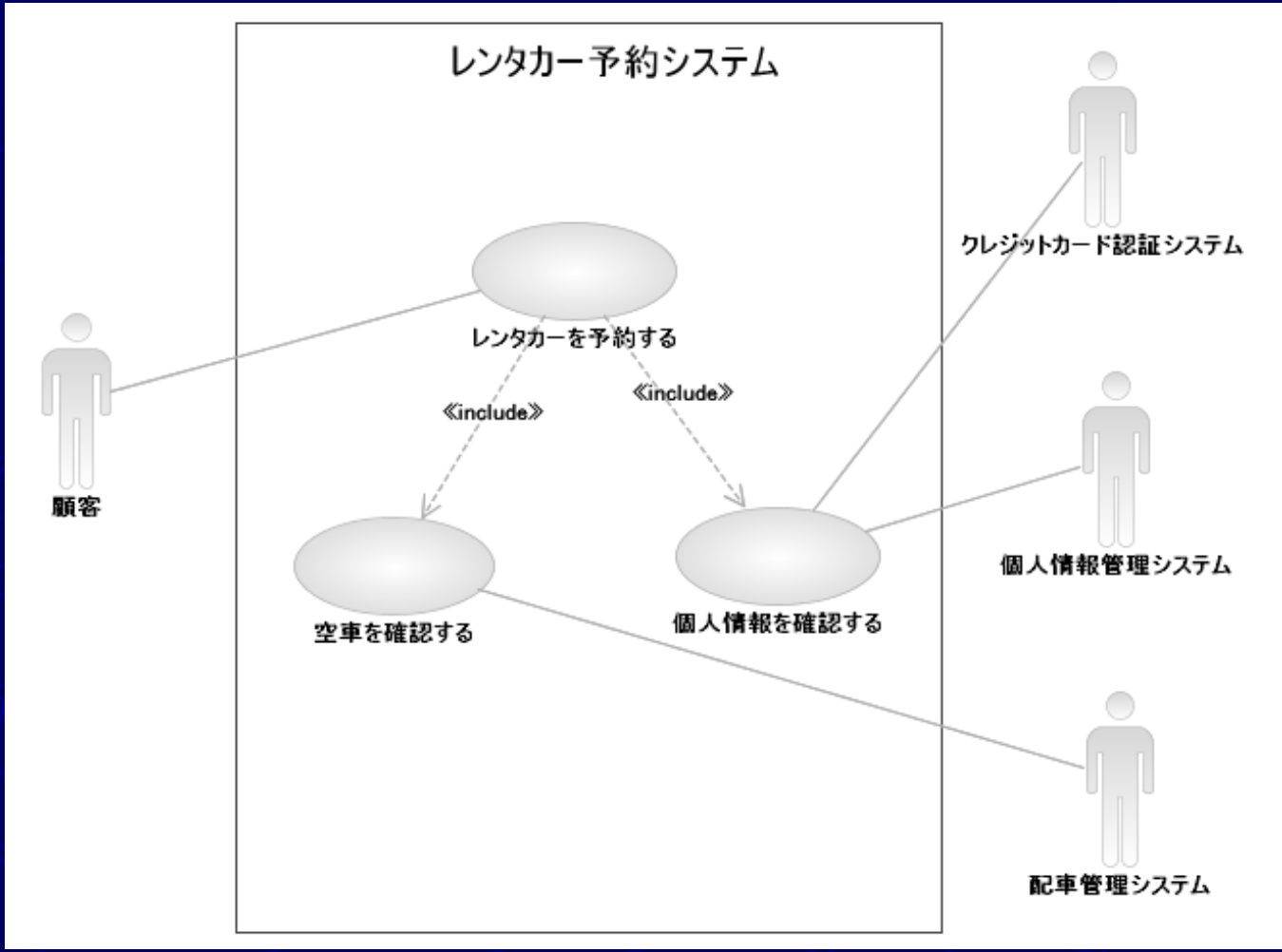
クラス図の例1



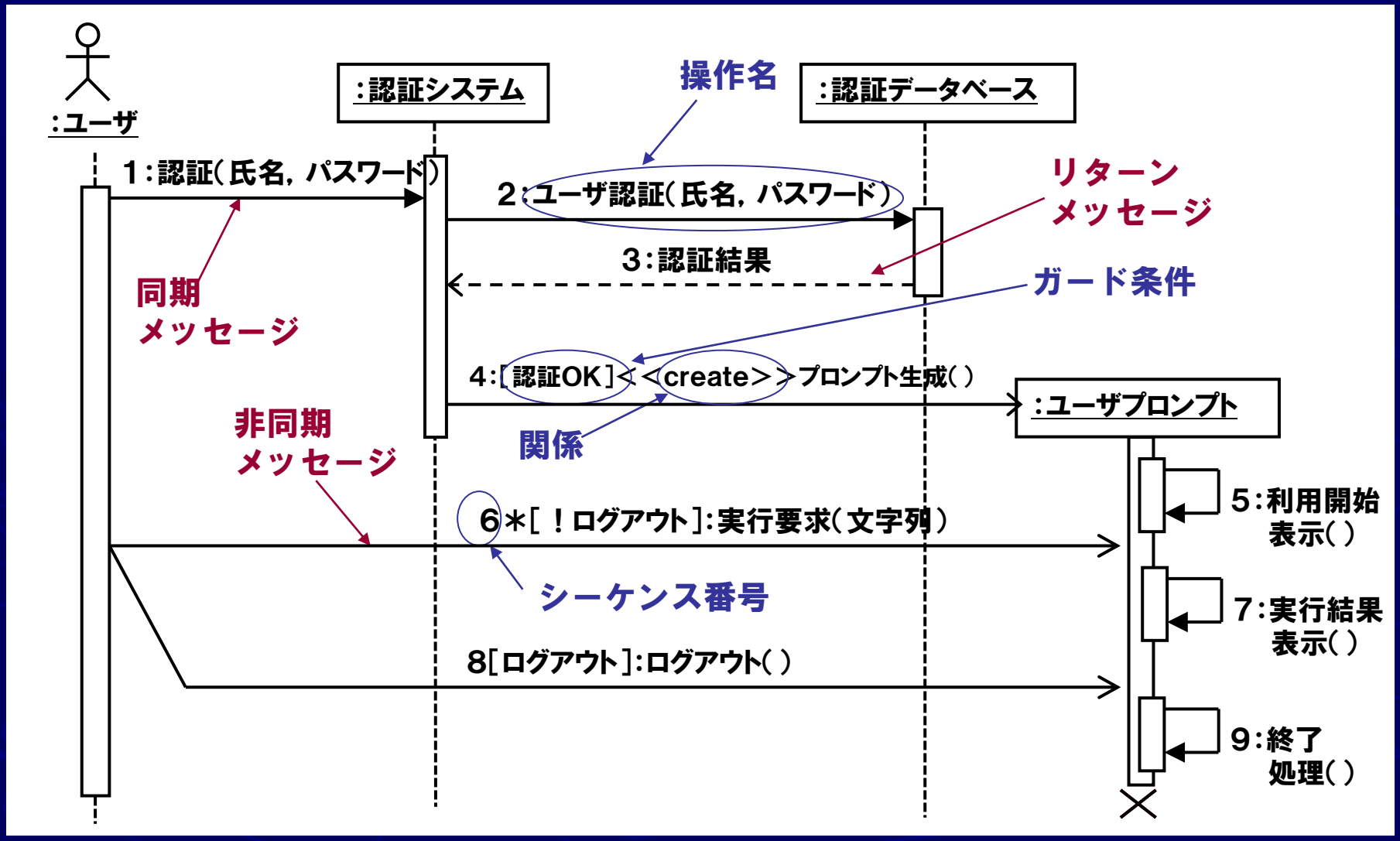
クラス図の例2


顧客
名前
住所
登録 ( )
変更 ( )
名前参照 ( )
住所参照 ( )
電話顧客
電話番号
登録 ( )
変更 ( )
電話番号参照 ( )
*
- 電話顧客
1..5
- 注文
発送
1
1
注文
- 発送
- 注文

ユースケース図の例


レンタカー予約システム
レンタカーを予約する
《include》
《include》
空車を確認する
個人情報を確認する
顧客
クレジットカード認証システム
個人情報管理システム
配車管理システム

シーケンス図の例


:ユーザ
:認証システム
操作名
:認証データベース
1:認証(氏名, パスワード)
2:ユーザ認証(氏名, パスワード)
リターン
メッセージ
3:認証結果
同期
メッセージ
ガード条件
4:[認証OK]<<create>>プロンプト生成()
:ユーザプロンプト
非同期
メッセージ
関係
6*[!ログアウト]:実行要求(文字列)
5:利用開始
表示()
シーケンス番号
7:実行結果
表示()
8[ログアウト]:ログアウト()
9:終了
処理()

まとめ：UML

- UML （Unified Modeling Language)
  - 統一モデリング言語
  - オブジェクト指向モデルを記述する言語
    - オブジェクト指向分析設計時に考えを具象化する道具
    - 技術者間のコミュニケーションの道具

- 各種図の利用の例
  - クラス図
  - ユースケース図
  - シーケンス図

Network Technologies

ネットワーク技術

通信プロトコルアーキテクチャ

- 通信プロトコルアーキテクチャ（プロトコルの階層図）
  - OSI基本参照モデル

7	アプリケーション層
6	プレゼンテーション層
5	セッション層
4	トランスポート層
3	ネットワーク層
2	データリンク層
1	物理層

OSI基本参照モデルの階層（１）

- 各層の詳細
  - 第１層（物理層）
    - 伝送路の物理的接続の確立や維持管理
    - 電気的条件や手順に関する特性の提供
    - ビット列伝送の保証
  - 第２層（データリンク層）
    - ケーブル内での複数ホストからのデータの競合（輻輳）の制御
    - 通信制御とエラー制御
  - 第３層（ネットワーク層）
    - データ交換の中継処理と経路制御
  - 第４層（トランスポート層）
    - 端末間同士の通信の保証
    - フロー制御（下位階層でのデータが輻輳しないように）

OSI基本参照モデルの階層（２）

第５層（セッション層）

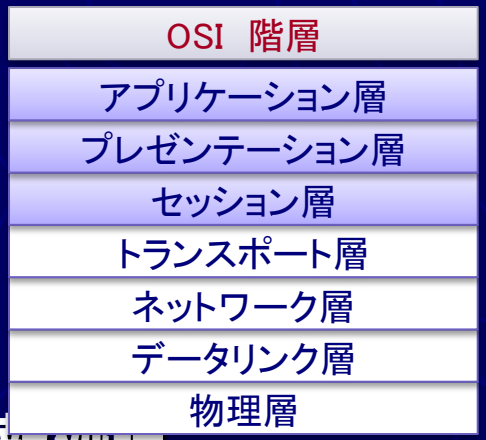
- 情報の伝送方法（半二重、全二重）の制御
- 同期処理

第６層（プレゼンテーション層）

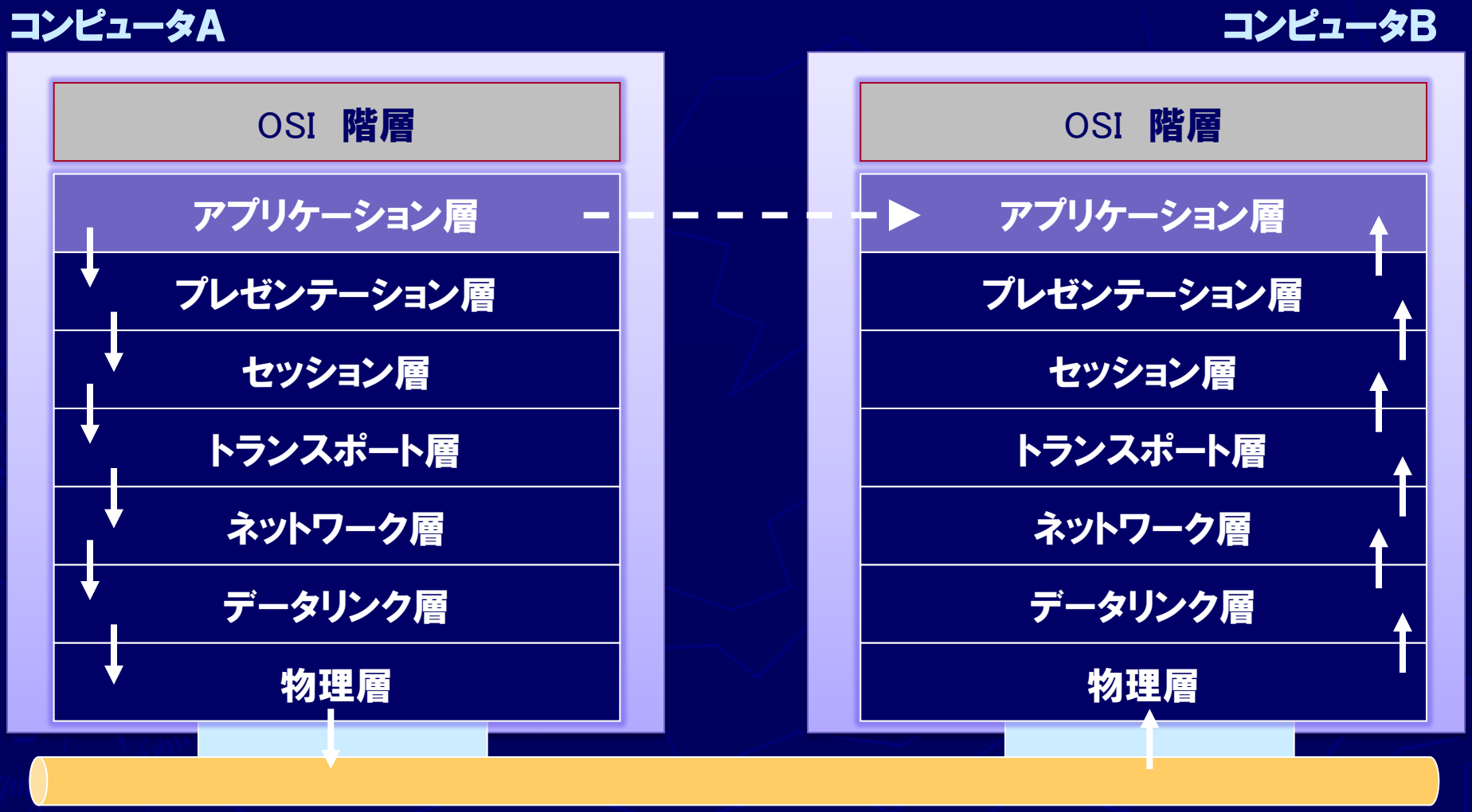
- コード変換、データの圧縮や暗号化などの情報処理

第７層（アプリケーション層）

- アプリケーション プログラムに対応するデータの処理単位を定義
- 各種情報サービスの実現



OSI基本参照モデルの階層（３）



TCP/IPプロトコルの概要

- TCP/IPプロトコル
  - 主に以下の二つのプロトコルを中心に構成
    - TCP（Transmission Control Protocol）
    - IP（Internet Protocol）
  - 事実上の業界標準プロトコル
    - デファクト・スタンダード
  - RFC（Request For Comments）によって規格化
    - http://www.rfc-editor.org/

TCP/IPの構成

- OSI階層モデルとTCP/IP階層モデル

ネットワークインターフェイス層

- ネットワークインターフェイス層（レイヤ１）
  - Network Interface Layer
  - OSIの物理層（第１層）とデータリンク層（第２層）に対応
  - ネットワークケーブルの物理的仕様
  - コネクタの形状
  - 各信号線の規格や意味
  - 隣接する機器とのデータ通信の方法
  - イーサネットを利用

インターネット層

- インターネット層（レイヤ２）
  - Internet Layer
  - OSIのネットワーク層（第３層）に対応
  - データの送受信者間の接続に関する規格
    - コンピュータの認識
    - 通信経路の選択

TCP/IP 階層	
4	アプリケーション層
3	トランスポート層
2	インターネット層
1	ネットワーク インターフェイス層

- プロトコル
  - IP（Internet Protocol）
  - ICMP（Internet Control Message Protocol）
  - ARP（Address Resolution Protocol）
  - OSPF（Open Shortest Path First）

Internet Protocol(１)

- IP(Internet Protocol)
  - 経路選択
  - コネクションレス型のデータグラム転送サービス
    - コネクションレス：データ転送開始前の初期接続不要

- 機能
  - 状態の管理と通信
    - IPによって行なわれる通信時の障害を検出
    - ICMPの機能を使って通信相手に通知
  - 経路選択
    - IPのネットワーク経路の選択

Internet Protocol(２)

- 機能(続き)
  - データの分割・組立
    - データグラムをネットワークインターフェイス層で定義されたデータ長に合わせ分割・組立
  - 伝送エラーチェック
    - IPヘッダの伝送エラーをチェック
  - パケットの滞留時間制御
    - ネットワーク内の滞留時間を監視
    - 規定値より大きくなったら削除

トランスポート層

- トランスポート層 （レイヤ３）
  - Transport Layper
  - OSI基本参照モデルのトランスポート層（第４層）に対応
  - ホスト間でのデータ通信に関わる規格
    - エンドツーエンド（End-to-End 通信）

- プロトコル
  - TCP（Transmission Control Protocol）
    - コネクション型通信プロトコル
  - UDP（User Datagram Protocol）
    - コネクションレス型通信プロトコル

TCP/IP 階層	
4	アプリケーション層
3	トランスポート層
2	インターネット層
1	ネットワーク インターフェイス層

TCPの機能

- 高信頼性・高安全性通信のために
  1. 基本データ送信（Basic Data Transfer）
     - セグメント単位でのデータ伝送
  2. 信頼性（Reliability）
  3. フロー制御（Flow Control）
  4. 多重化（Multiplexing）
  5. コネクション（Connection）

TCPの機能(2．信頼性)

- 信頼性
  - データへの通し番号(Sequential Number)の添付
  - 肯定応答(Positive Acknowledge)方式を利用したデータの送受信

- エラー回復
  - データの紛失・重複の障害回復
  - データ送信後、一定時間内に応答がない場合は再送
- データ転送の信頼性の保証
  - 転送シーケンス制御によりデータ抜けのチェックを行なう
  - 双方向データ転送(ストリーム型)をサポート

TCPの機能(３．フロー制御)

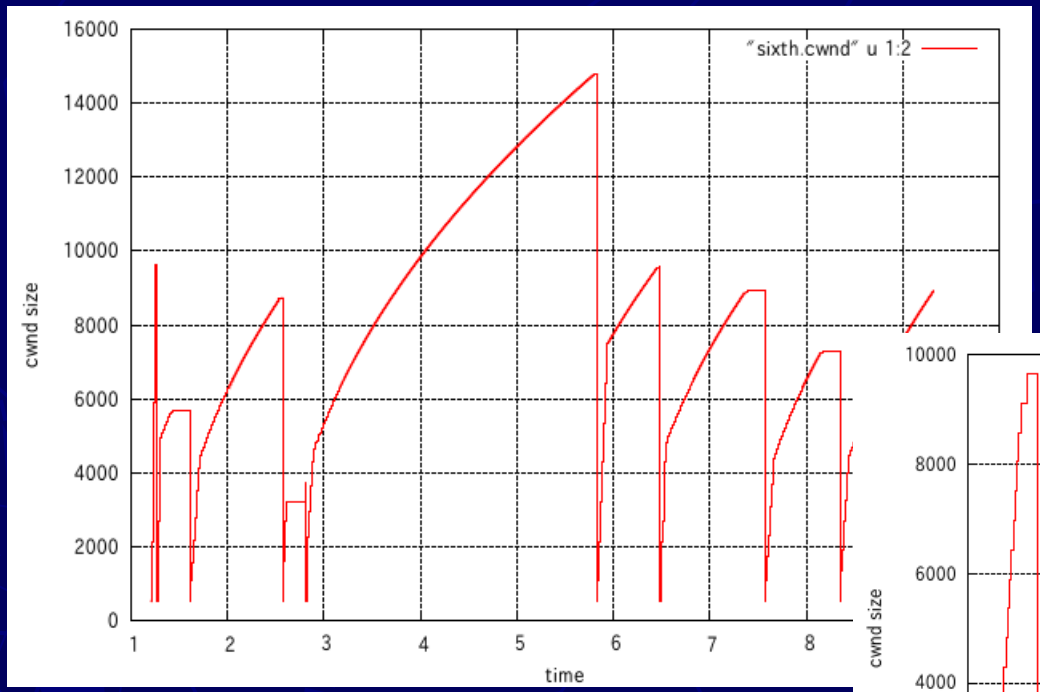
- ウィンドウによるフロー制御
  - ウィンドウサイズを利用したフロー制御

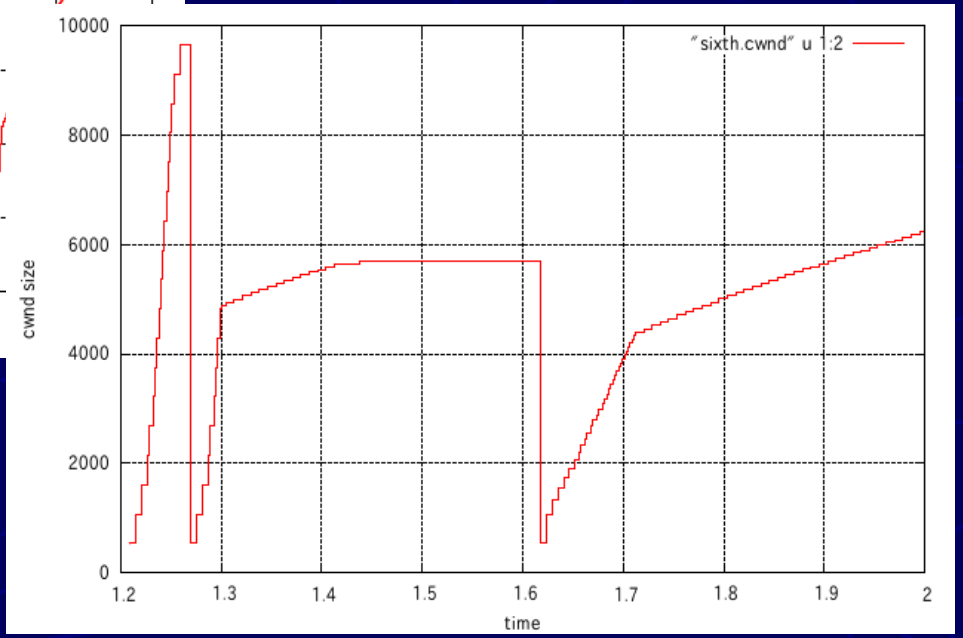
- フロー制御(データ流量の制御)
  - 送信データ量を調整・送受信側の同期

- ウィンドウ制御
  - 受信側がウィンドウ値(受信側が受信可能なデータ量)を送信側に通知
    - 送信側はウィンドウ値までは受信側からの応答を待たずに送信可

TCPの機能(3. フロー制御)

- Congestion Window





TCPの機能(4. 多重化)(1)

- 多重化
  - ポートを利用した通信
  - ソケット通信の導入

- ソケット通信(利用者インターフェイス)
  - アプリケーション・プログラム・インターフェイス
  - ソケットを利用し、TCP/IPの通信をプログラム
  - ポート番号によってアプリケーションを識別

TCPの機能(4. 多重化)(2)

- ポート番号
  - RFC1700 (ASSIGNED NUMBERS) で割り当て
  - Well Known Ports: 0 - 1023
    - UNIX Standard Service : 256 - 1023
  - Registered Ports: 1024 - 49151
  - Dynamic and/or Private Ports: 49152 - 65535
    - 利用者が任意に利用可能

http://www.iana.org/
assignments/port-numbers

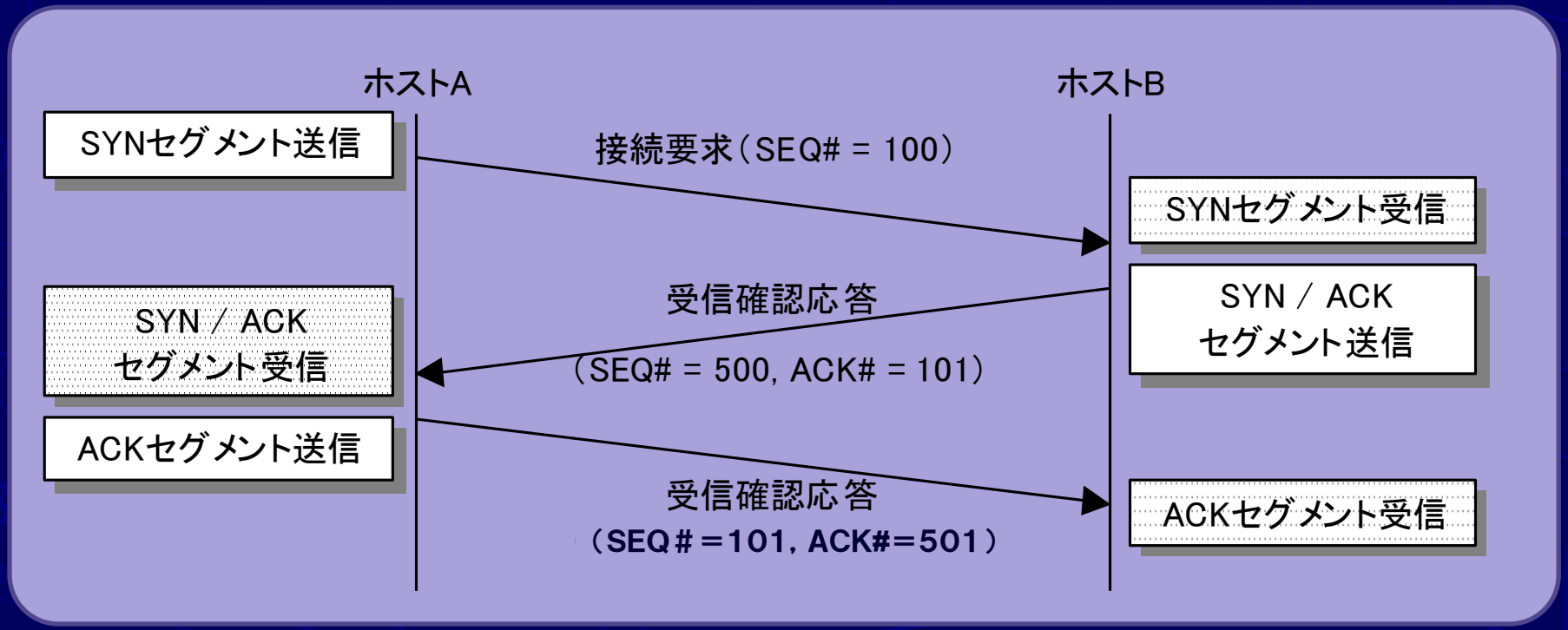
TCPの機能(５．コネクション)

- コネクション
  - ハンドシェイク(Handshake)方式を利用した接続の確立・解除(切断)

- トランスポート・コネクションの確立・解除(切断)
  - 通信開始時に送信側と受信側との間に論理的な通信路(コネクション)を設定
  - 通信終了時に設定されていた通信路を解除(切断)

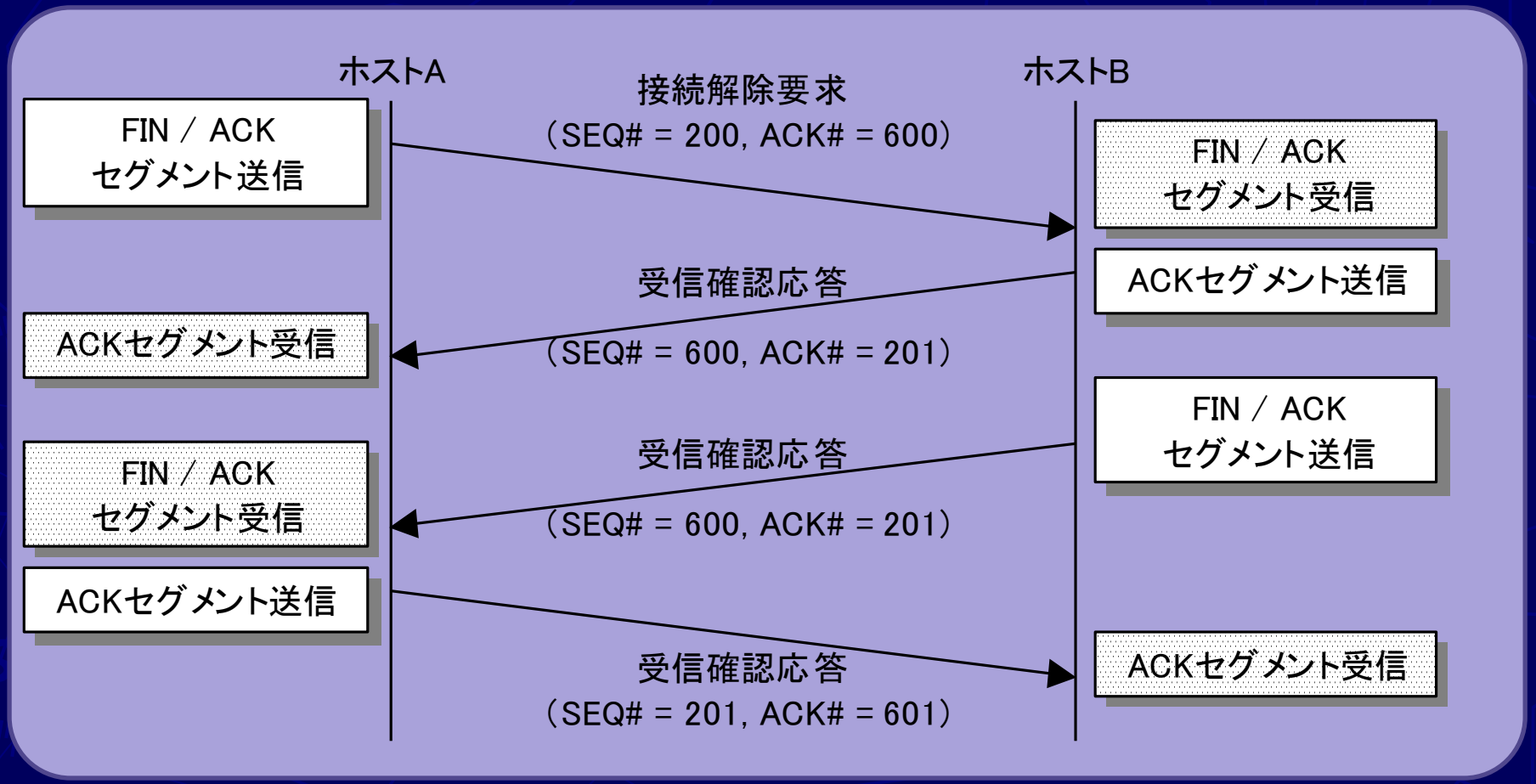
コネクションの確立

- トランスポート・コネクションの確立
  - 通信開始手順
    - Three Way Handshake



コネクションの解除（切断）

- トランスポート・コネクションの解除（切断）



UDPとは

- UDP（User Datagram Protocol）
  - コネクションレス型データグラム転送
    - 転送開始時に接続手続きを行なわない
  - 伝送の簡易制御方式
    - データ到着の順序制御などのシーケンス制御やエラー制御を行なわない
    - 上位層からのデータは分割せずに送信
  - 低負荷伝送
    - 構造が簡単、かつ、コネクションレス型転送なので、転送効率に優れる

まとめ：ネットワーク技術

- OSI基本参照モデル
- TCP/IPモデル
  - ネットワークインターフェイス層
  - インターネット層
  - トランスポート層
  - アプリケーション層
- トランスポート層
  - TCP：コネクション型プロトコル
  - UDP：コネクションレス型プロトコル