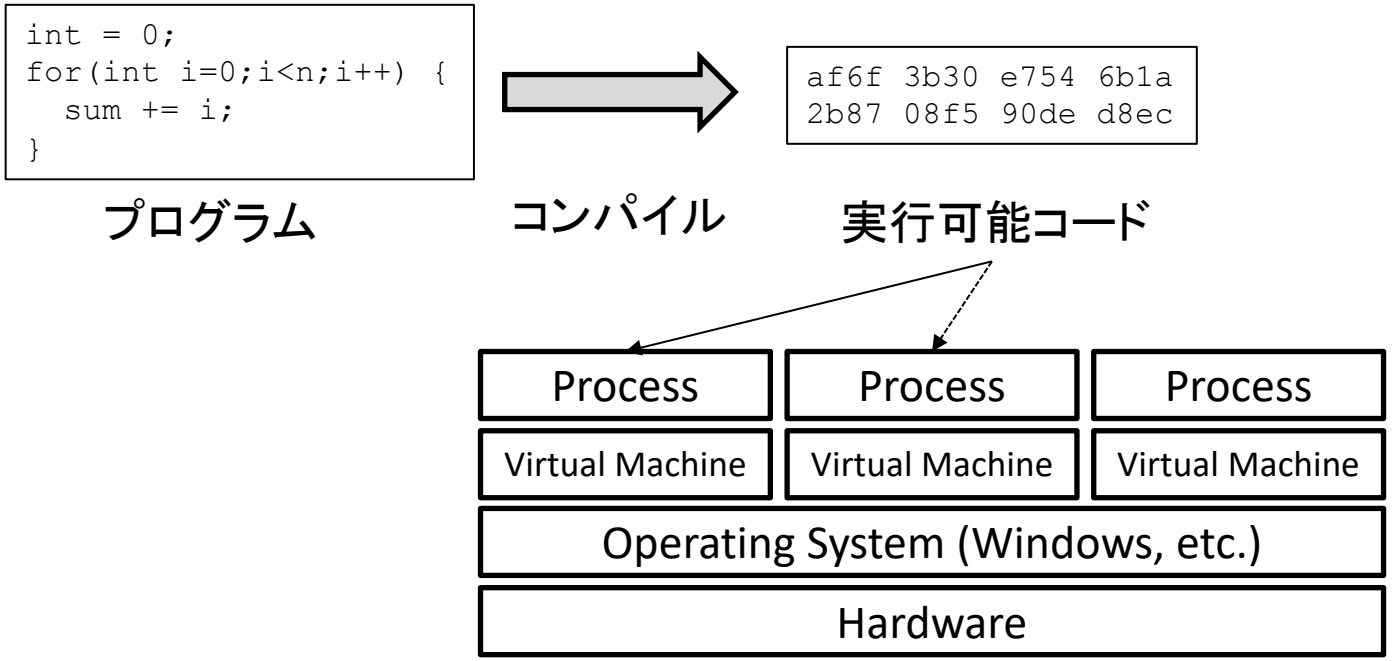
分散システム

第3回 プロセス

双見 京介（FUTAMI Kyosuke）
高田秀志（TAKADA Hideyuki）
2025年10月

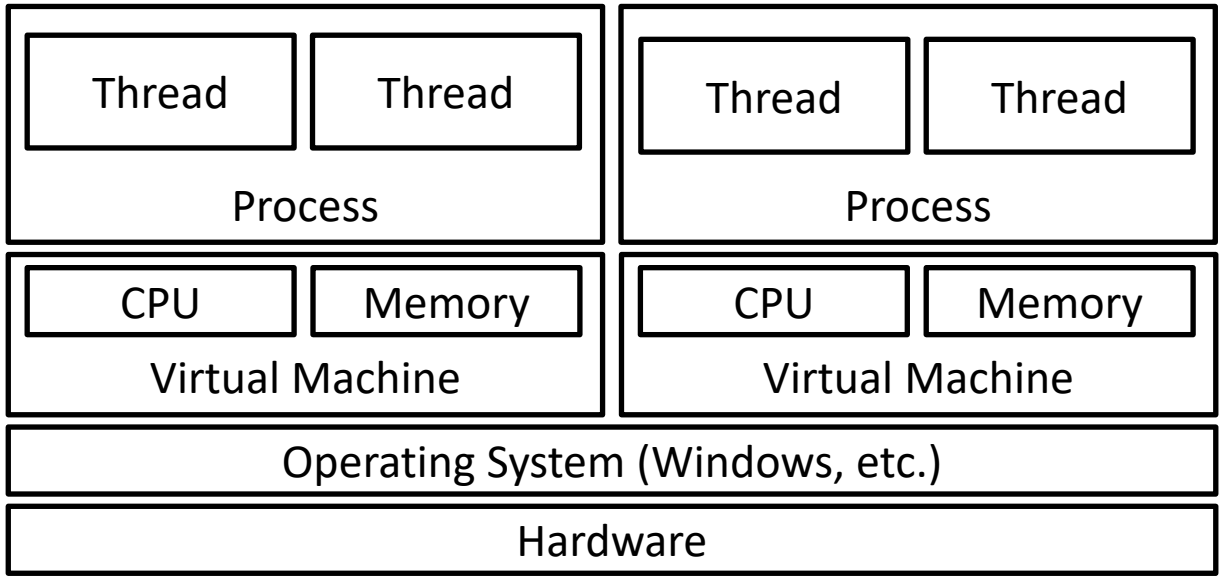
プロセス（Process）

- 実行可能コードは「プロセス」として実行される
- 一つの実行可能コードを複数のプロセスで同時に実行することができる
- 複数のプロセスによるハードウェア資源（CPU，メモリなど）の共有はOSによって管理される



スレッド（Thread）

- 実行されているプログラムの部分を時系列に沿って「筋」として捉えたもの
- 一つのプロセスの中で，複数のスレッドを実行可能
- プロセス内のスレッドはCPUやメモリを共有
- 一つのスレッドがブロックされても（サーバからの応答待ちなど），プロセス全体としてブロックされることを防ぐことが可能
- マルチプロセッサシステム上では並列性を活かすことが可能



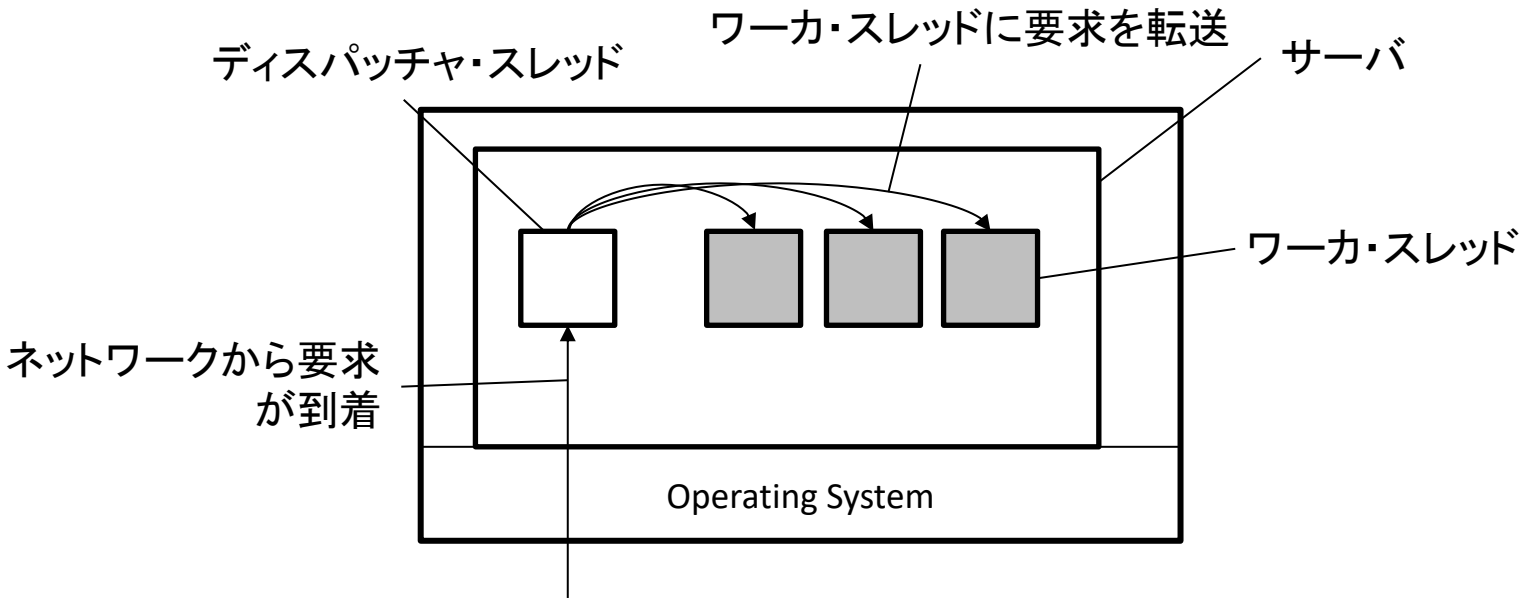
クライアントにおけるマルチスレッド

- Webブラウザはマルチスレッドにより実行されている
- それぞれのスレッドは，サーバからのHTMLファイルの取得やユーザインタラクション（マウス操作等）の処理等，個別の役割を果たす

- テキストや画像など，複数のコンテンツがサーバから並列に取得される
- コンテンツの取得中（サーバとの通信中）でも，ユーザはページに対する操作（スクロールなど）が可能

←ロード中のWebページ

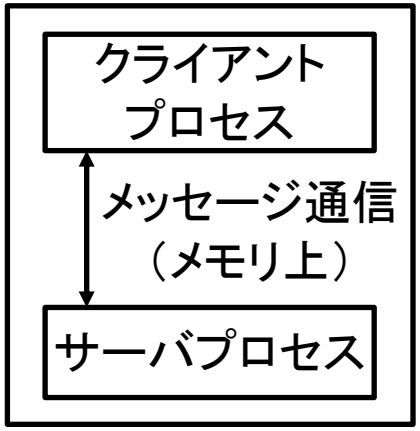
サーバにおけるマルチスレッド



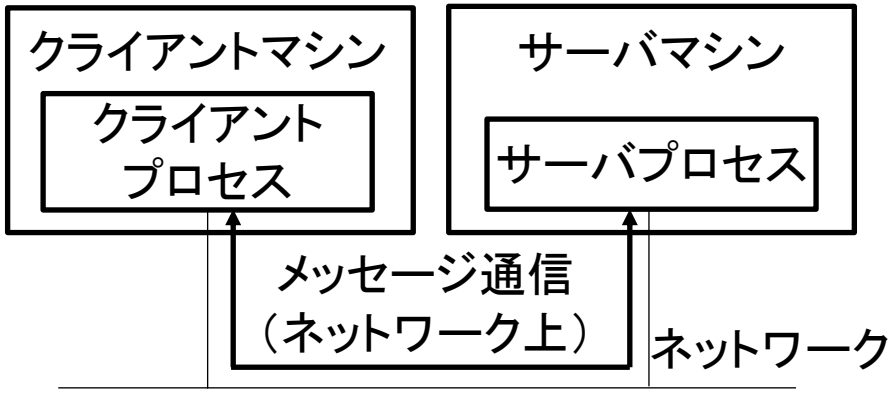
- ディスパッチャがネットワークから到着する要求を取得
- クライアントは，サーバのwell-knownポート（例えばHTTPは80）へ要求を送信
- サーバは，処理可能なワーカ・スレッドを選択し，要求を転送

クライアントサーバモデル

- 分散システムで最も広く採用されている構成の一つ
- サービスを提供する「サーバ」と，サービスを利用する「クライアント」に分け，それぞれを別のプロセスとして実現
- サーバプロセスとクライアントプロセスはメッセージ通信を行う

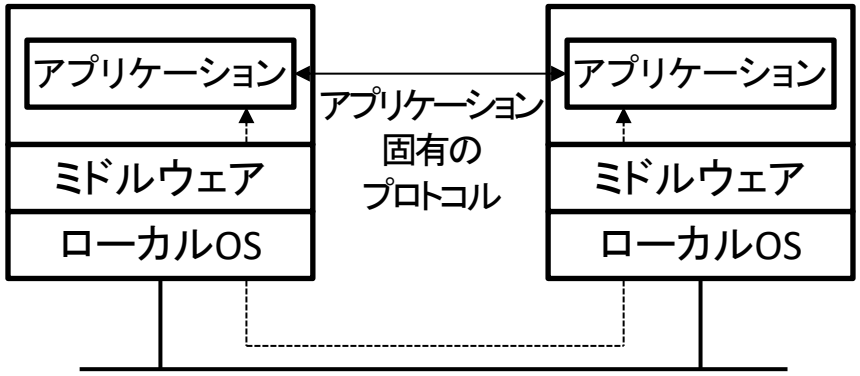


同一マシン内での実装

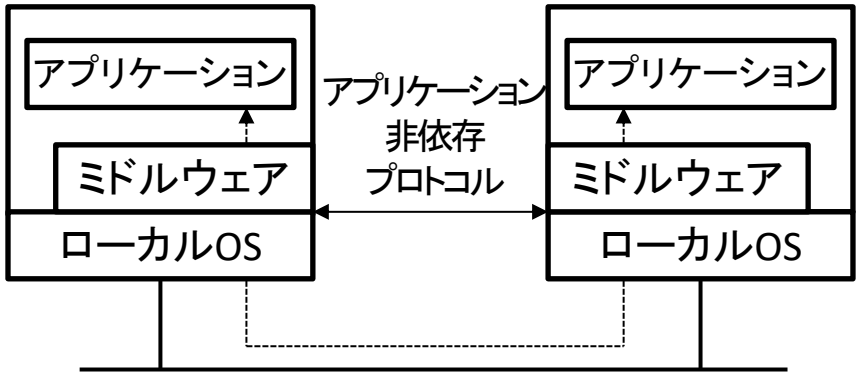


異なるマシン内での実装

クライアントの種別

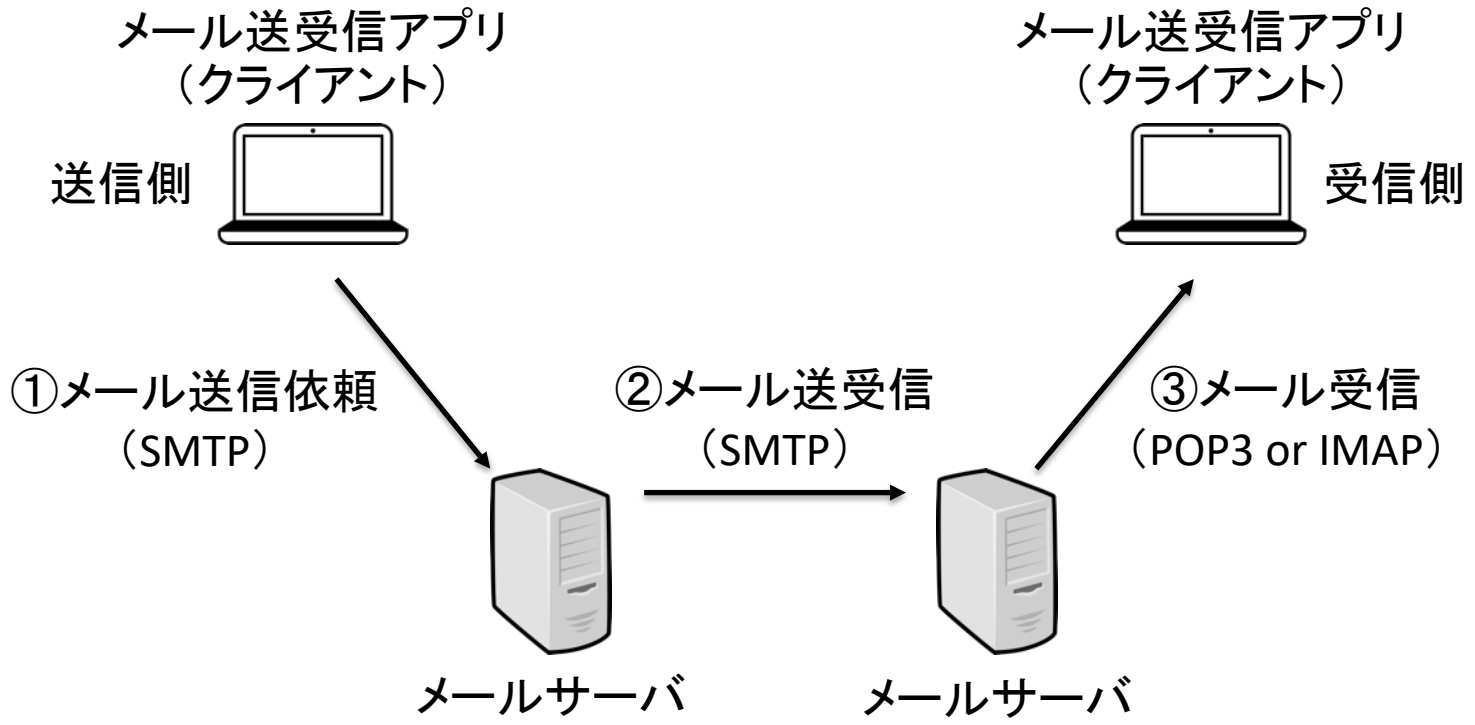


- アプリケーションごとに固有のサーバ・クライアントの組み合わせを持つ（カレンダーアプリ，お天気アプリなど）
- アプリケーション層のプロトコルにより，サーバプロセスとのやりとりを行う



- クライアントマシンはサービスに対するユーザインタフェースのみを提供し，ローカルなストレージを持たない「端末」として動作（**シンクライアント**）
- **Xウィンドウシステム**がよく知られた実現例

電子メールシステム

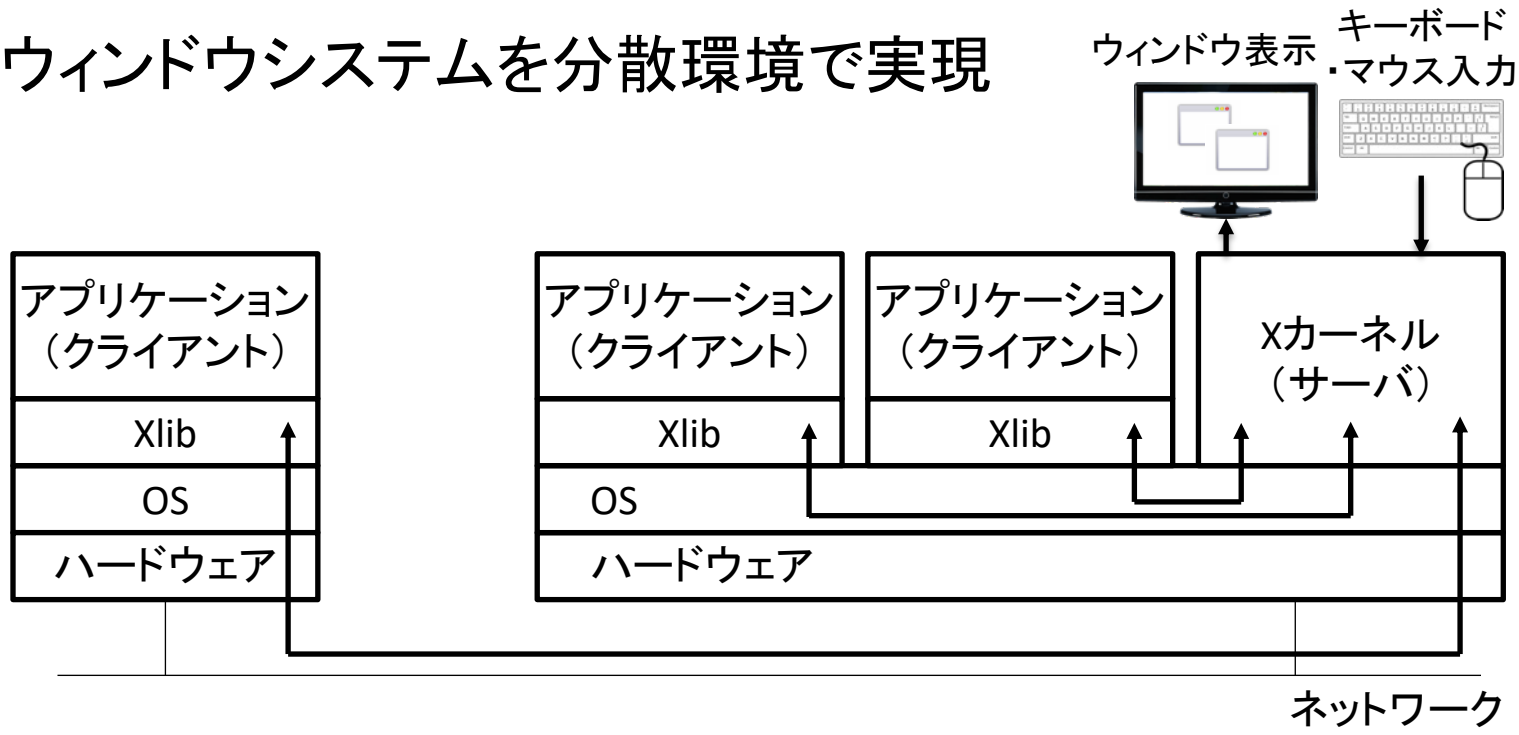


サーバとクライアント間は，電子メールシステム固有のプロトコルで通信

- SMTP (Simple Mail Transfer Protocol)
- POP3 (Post Office Protocol 3)
- IMAP (Internet Message Access Protocol)

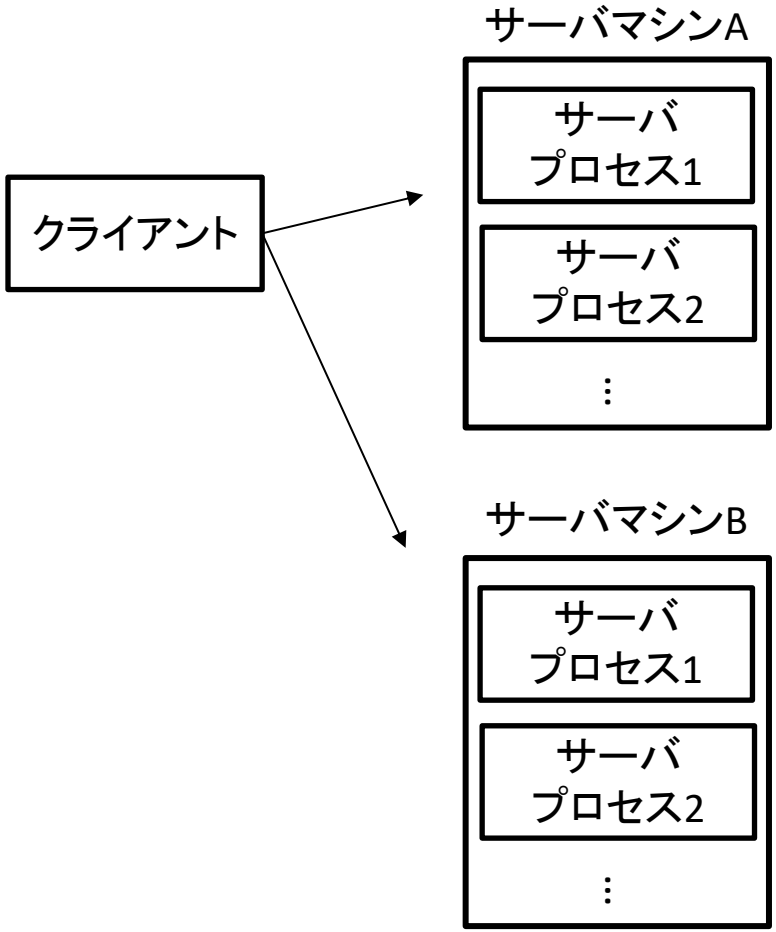
Xウィンドウシステム

ウィンドウシステムを分散環境で実現



- ユーザが利用する端末のXカーネル（サーバ）は，ユーザインタフェースを提供
- アプリケーションは提供する機能（お絵かき等）に応じた処理を実現
- クライアント側のXlibとサーバ側のXカーネルは，アプリケーションに依存しないプロトコルで通信（描画命令，マウスイベント，キー入力イベント等）

サーバの構成



- サーバマシンは
「アドレス」により識別
- サーバプロセスは
「ポート番号」により識別

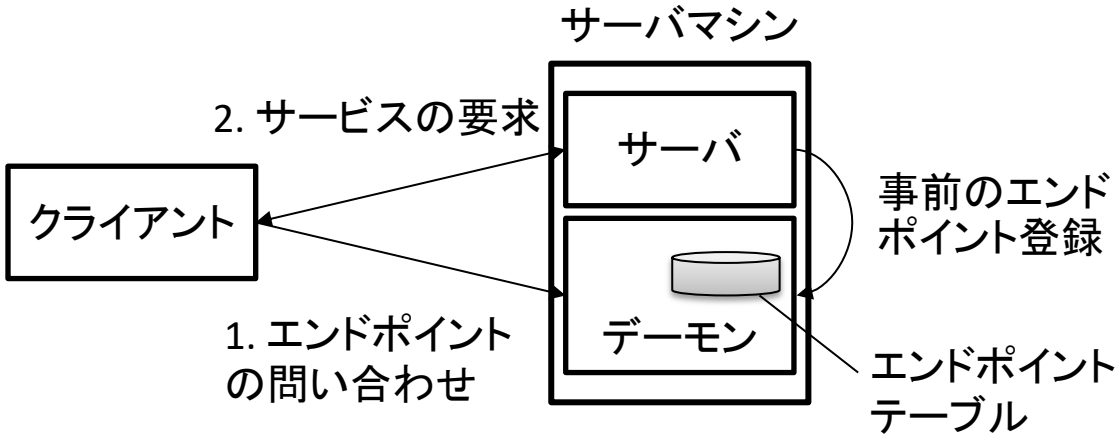
IPアドレス: 133.19.170.14
HTTP（Webサーバ）のポート番号: 80
SMTP（メール送信サーバ）のポート番号: 25

よく使われるプロトコルのポート番号は，
共通の申し合わせによって決まっている。

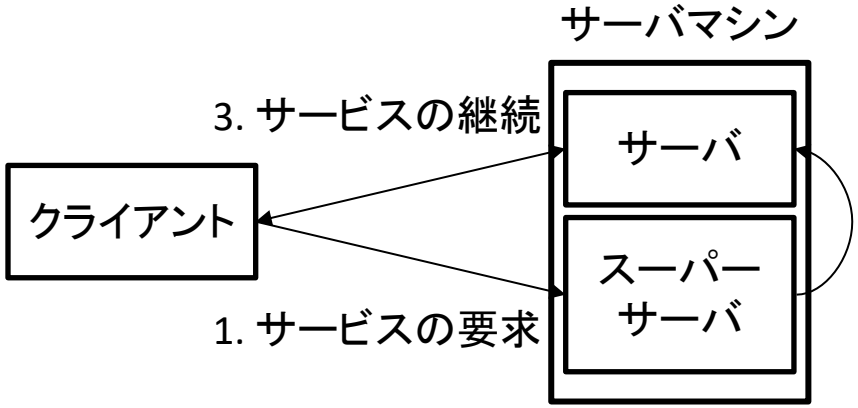
サーバの種類

- 処理の主体
  - 反復（Iterative）サーバ
    一つのプロセスがクライアントからの要求を繰り返して処理
  - 並行（Concurrent）サーバ
    クライアントからの要求を受け付けたプロセスが，他のプロセスに処理を転送
- エンドポイントの解決（次ページ）
- 状態の保持
  - ステートレス（Stateless）サーバ
    - クライアントの状態に関する情報を保持しない
    - クライアントの状態によってサーバの振る舞いは変わらない
    - 水平分散の実現が容易
  - ステートフル（Stateful）サーバ
    - クライアントの状態に関する情報を持続的に保持
    - サーバとクライアント間でやりとりする情報の量が減少
    - 水平分散や障害復旧処理が複雑化

エンドポイントの解決

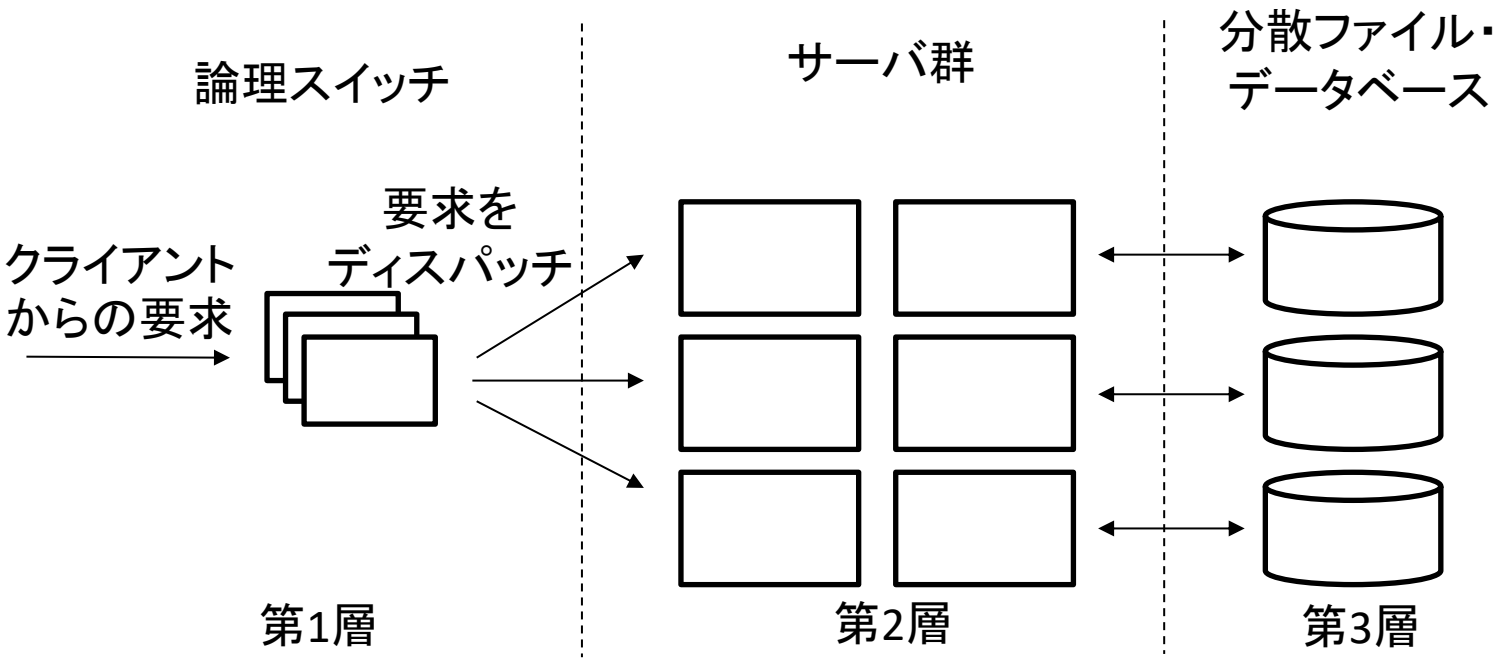


クライアントがデーモンに
エンドポイントを問い合わせ



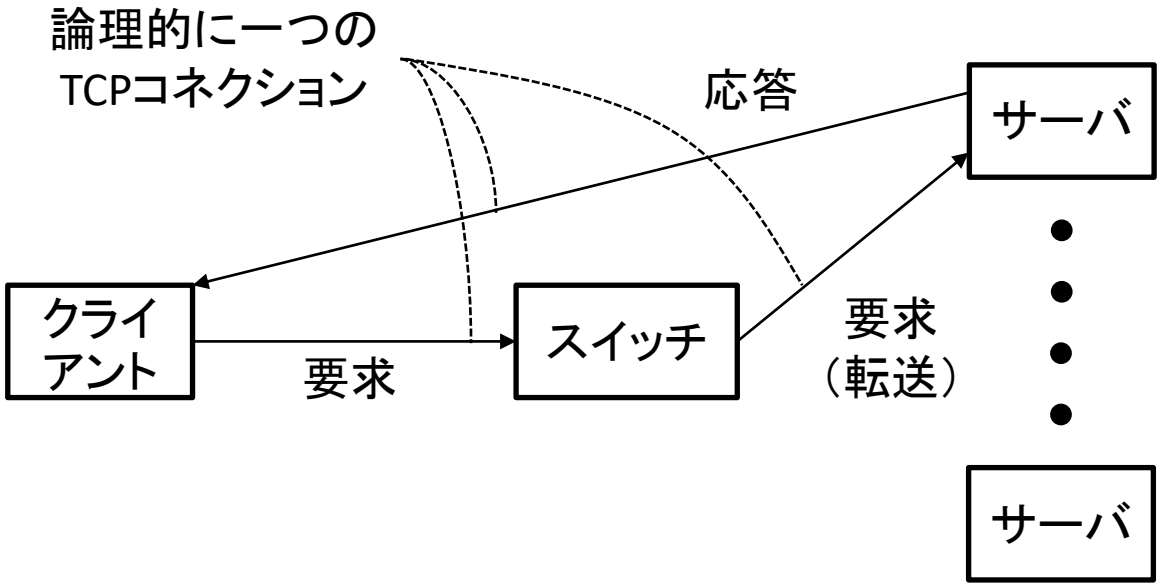
スーパーサーバがクライア
ントの要求に応じたサーバ
プロセスを生成

サーバクラスタ



- サーバクラスタでは，高帯域・低遅延のネットワークで複数のマシンを接続
- 第1層の実現方式
  - TCPハンドオフ（トランスポート層）
  - コンテンツアウェア要求分散（アプリケーション層）

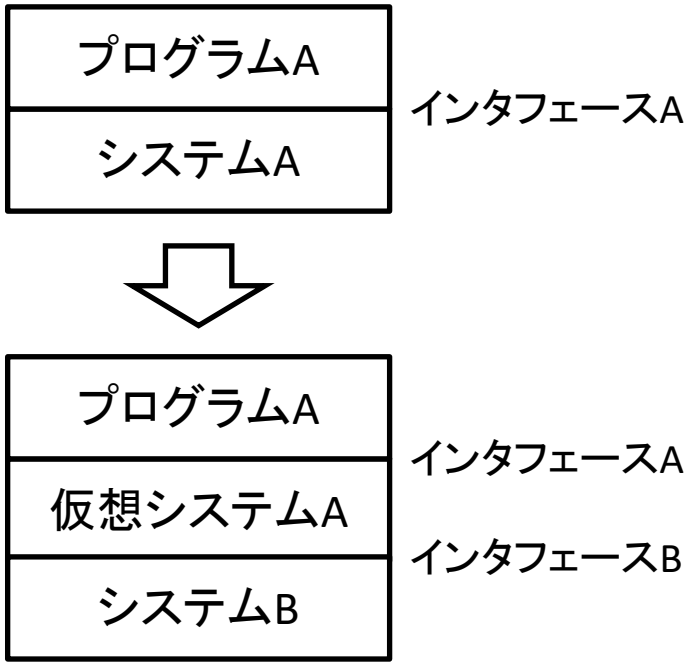
TCPハンドオフの原理



- スイッチが要求パケットを最も適切なサーバに転送
- サーバは，スイッチのIPアドレスをヘッダに挿入してACKメッセージを送信

仮想化（Virtualization）

- 仮想化の概念
  - アプリケーションプログラムなどの上位層に仮想的なインタフェースを提供
  - システムAが提供するインタフェースA上で動作するプログラムAをシステムBで動作させるために，システムB上にインタフェースAを提供する仮想システムAを構築
  - 「無いものを有るように見せる」
    （↔「有るものを無いように見せる」のは「透明化」）
- 仮想化の目的
  - 新しいプラットフォーム上でのプログラムの再利用
  - プラットフォームの多様性の削減
  - 需要に応じた計算資源の割り当て

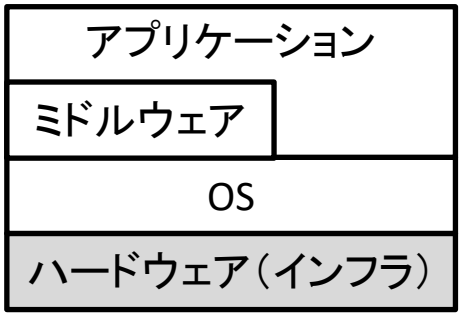


クラウドコンピューティング

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

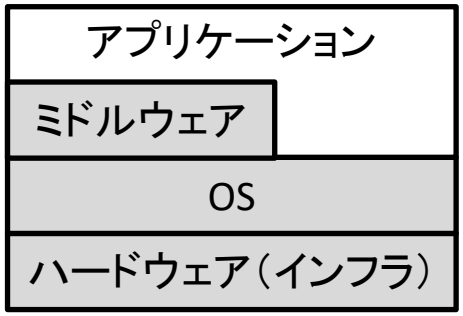
(National Institute of Standards and Technology)

ハードウェア環境を提供
（ユーザはまずOSを
インストール）



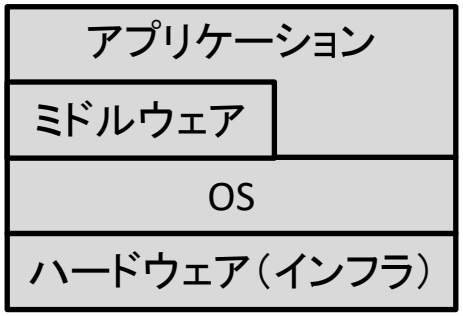
Infrastructure as
a Service
(IaaS)

OSやDB環境を提供
（ユーザはアプリケーションを開発）



Platform
as a Service
(PaaS)

アプリケーションを提供
（ユーザはメールサービス
等を利用）



Software
as a Service
(SaaS)

コードマイグレーション

- プログラムコード自体が分散システム内を移動
  - 負荷の高いコンピュータから負荷の低いコンピュータへプロセスを移動
  - サーバに大量のデータをアクセスしているクライアントをサーバへ移動
  - サーバからクライアントへプログラムをダウンロードして実行
- 移動性（mobility）
  - プログラムと初期値が移動先に渡されて実行（弱移動性）
  - 実行途中のプロセスを中断し，スナップショット（データやプログラムカウンタ等）が移動先に渡されて再開（強移動性）
- プログラム起動（initiation）
  - コードを送り出す側がプログラムを起動（送信者開始型）
  - コードを受け取る側がプログラムを起動（受信者開始型）