分散システム

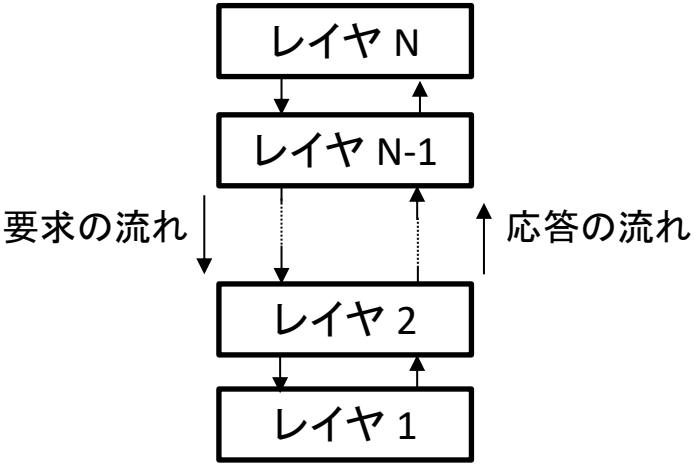
第2回 アーキテクチャ

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

アーキテクチャ

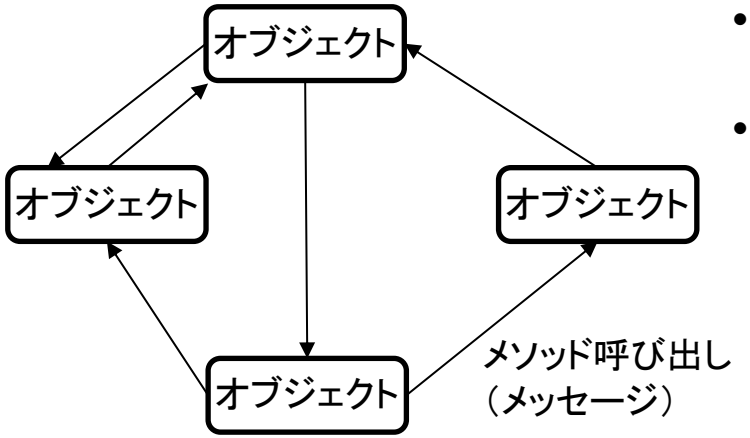
- 以下の項目を形式化したもの
  - 分散システムを構成するソフトウェアコンポーネント間を接続する方法
  - コンポーネント間のデータ交換
  - 単一のシステムとして振る舞うための構成
- コンポーネントは，明確な定義，要求，実装を伴ったインタフェースを持ち，コンポーネント単位で置き換え可能
- 以下の4つを取り上げる
  - 階層型アーキテクチャ
  - オブジェクトベースアーキテクチャ
  - データ中心型アーキテクチャ
  - イベントベースアーキテクチャ

アーキテクチャ型（1）



(a) 階層型アーキテクチャ

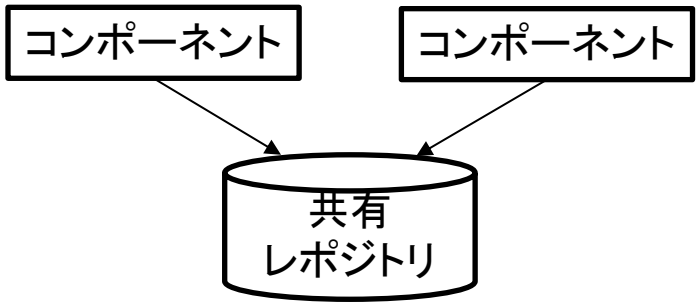
- レイヤ L_i のコンポーネントは下位レイヤ L_{i-1} のコンポーネントを呼び出し可能
- レイヤからレイヤへ制御が流れる
  - 要求は階層を下方向へ
  - 応答は階層を上方向へ



(b) オブジェクトベースアーキテクチャ

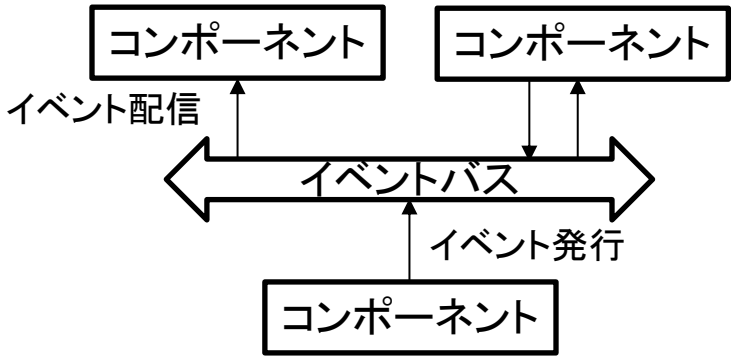
- オブジェクト（システムを構成するコンポーネント）は「遠隔手続き呼び出し」によって接続
- クライアント・サーバアーキテクチャに符合

アーキテクチャ型（2）



(c) データ中心型アーキテクチャ

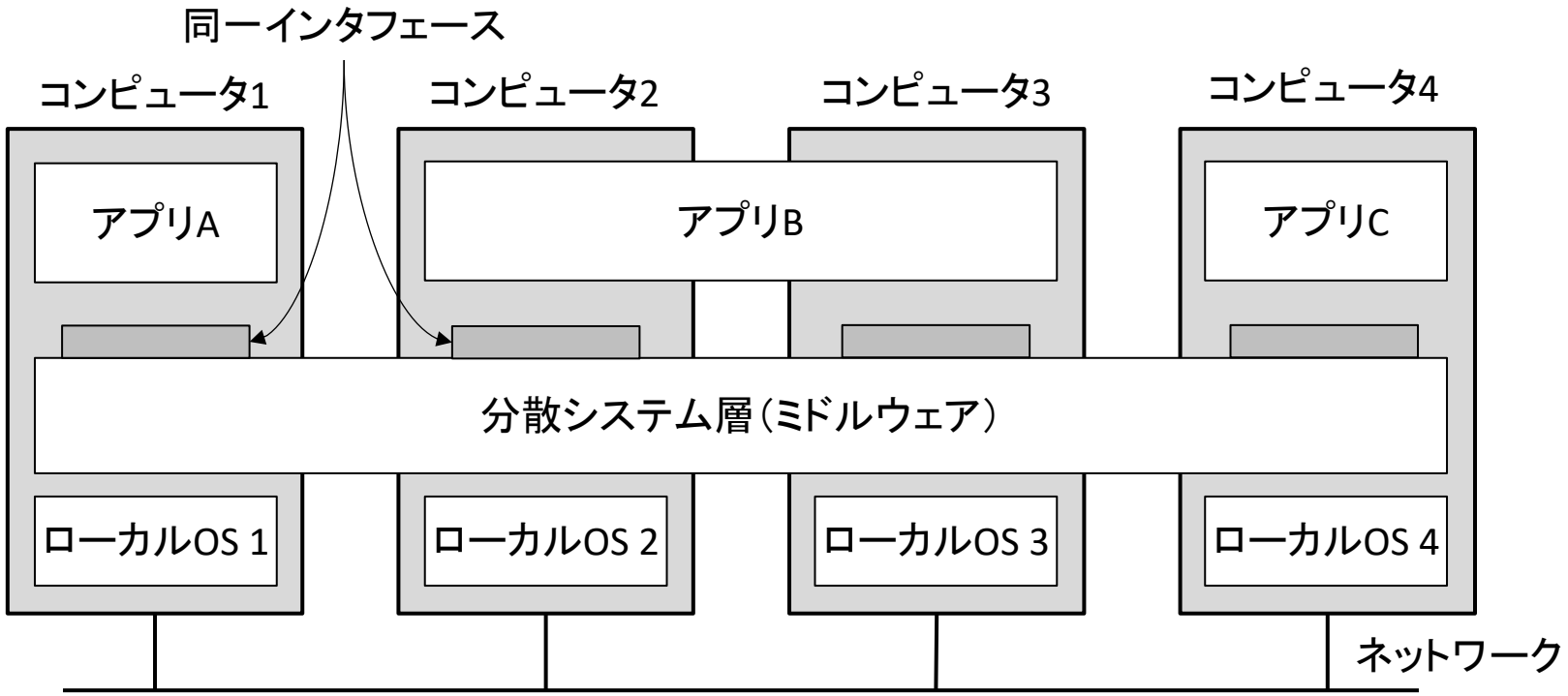
- コンポーネントは共有レポジトリを介して通信
- 例として，分散共有ファイルシステム，共有Webベースサービスなど



(d) イベントベースアーキテクチャ

- 出版・購読システム（publisher/subscriber system）と呼ばれる
- コンポーネントはイベント伝播により通信
- イベントの配信を受けたいコンポーネントはイベントを「購読」，イベントを発行したいコンポーネントはイベントを「発行」

ミドルウェア（Middleware）



- ハードウェアやローカルOSに依らず，ミドルウェアによって同一インタフェースをアプリケーションに提供
- アプリケーション間通信，セキュリティ機能，障害対策機能などを提供

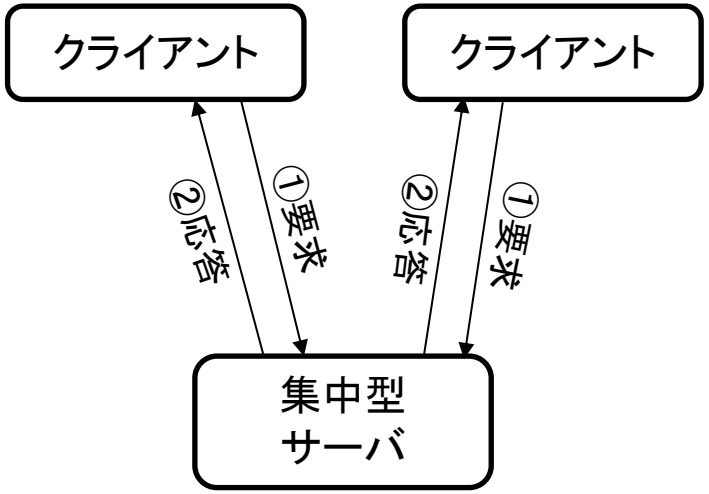
システムアーキテクチャ

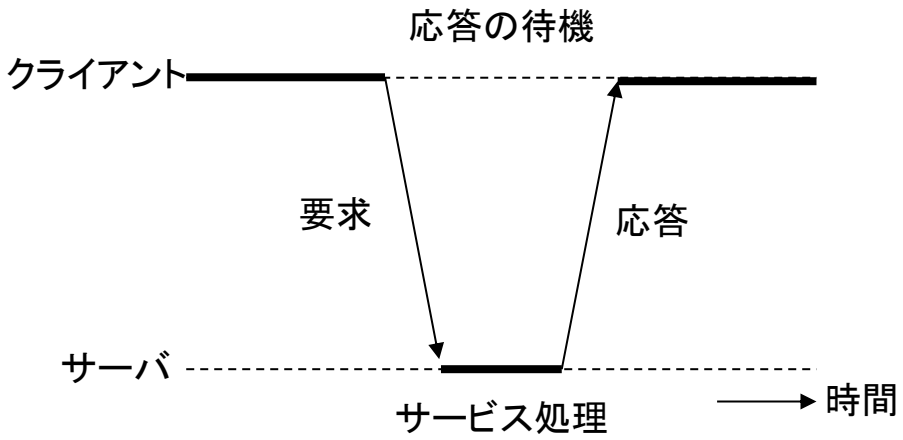
- ソフトウェアコンポーネント群，それらの間の対話，配置場所を規定

- 3つの構成
  - 集中型（Centralized）アーキテクチャ
  - 非集中型（Decentralized）アーキテクチャ
  - ハイブリッド（Hybrid）アーキテクチャ

集中型アーキテクチャ

- 「クライアント」と「サーバ」の2つのプロセスから構成（クライアント・サーバモデル）
- 通常，多くのクライアントが単一の集中型サーバを共有

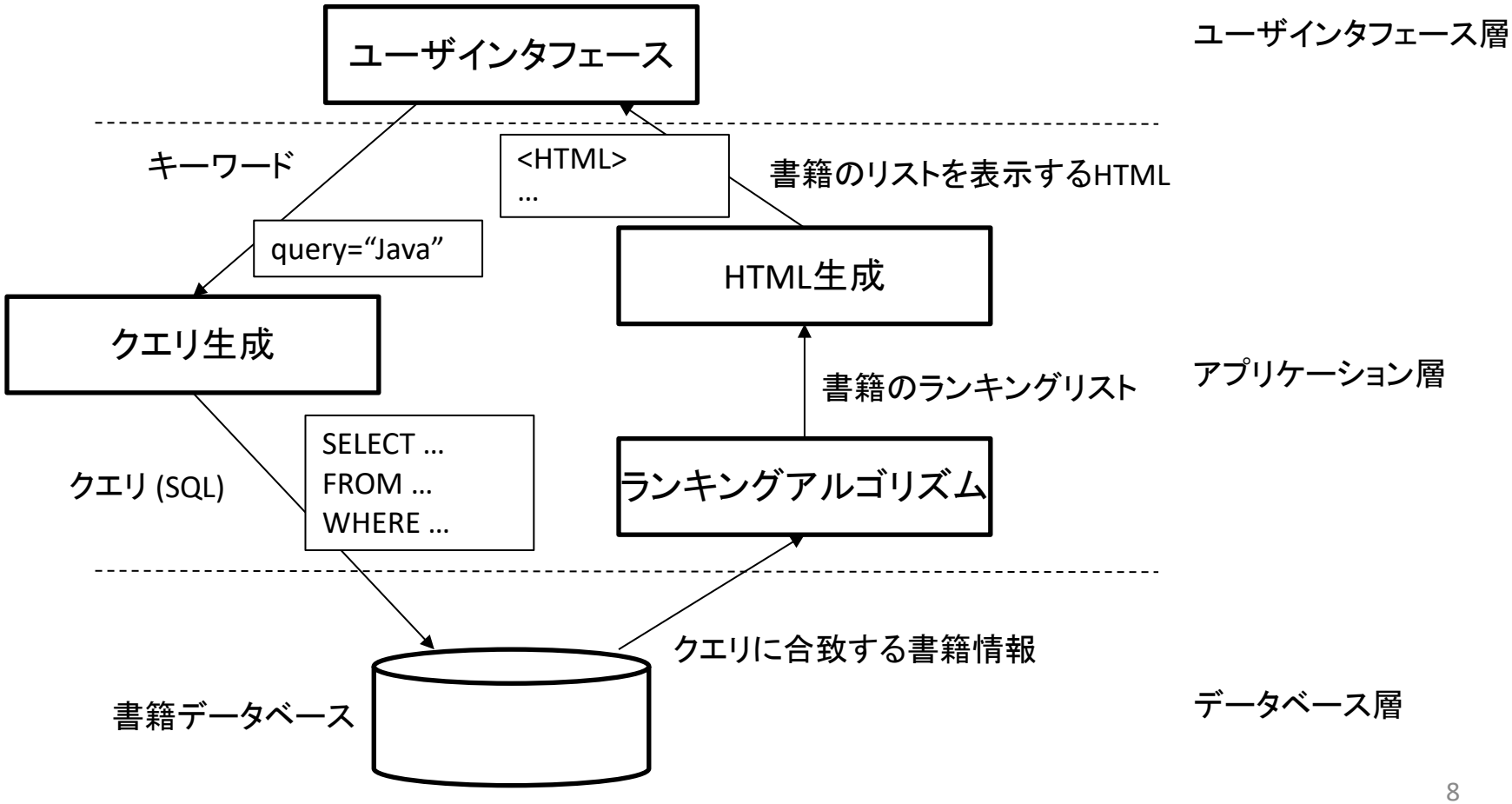




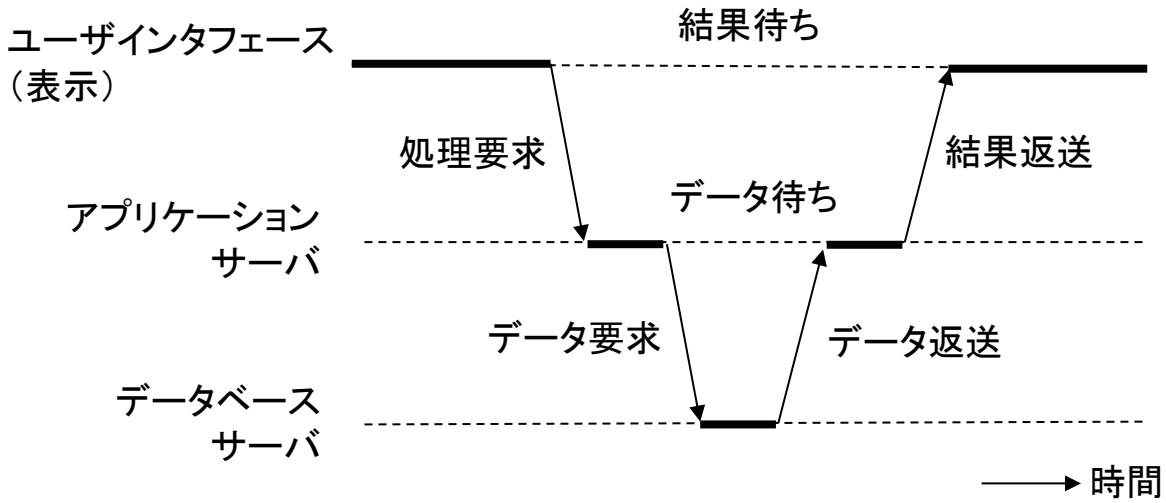
- サーバはクライアントからの要求を待つ
- クライアントは要求の送信後，サーバからの応答を待つ（ブロックされる）

アプリケーションのレイヤ化

- データベースアクセスを伴うシステムの典型的な構造
- ユーザインタフェース層はクライアント，アプリケーション層はクライアントとサーバの双方，データベース層はサーバとして振る舞う



3層クライアントサーバ

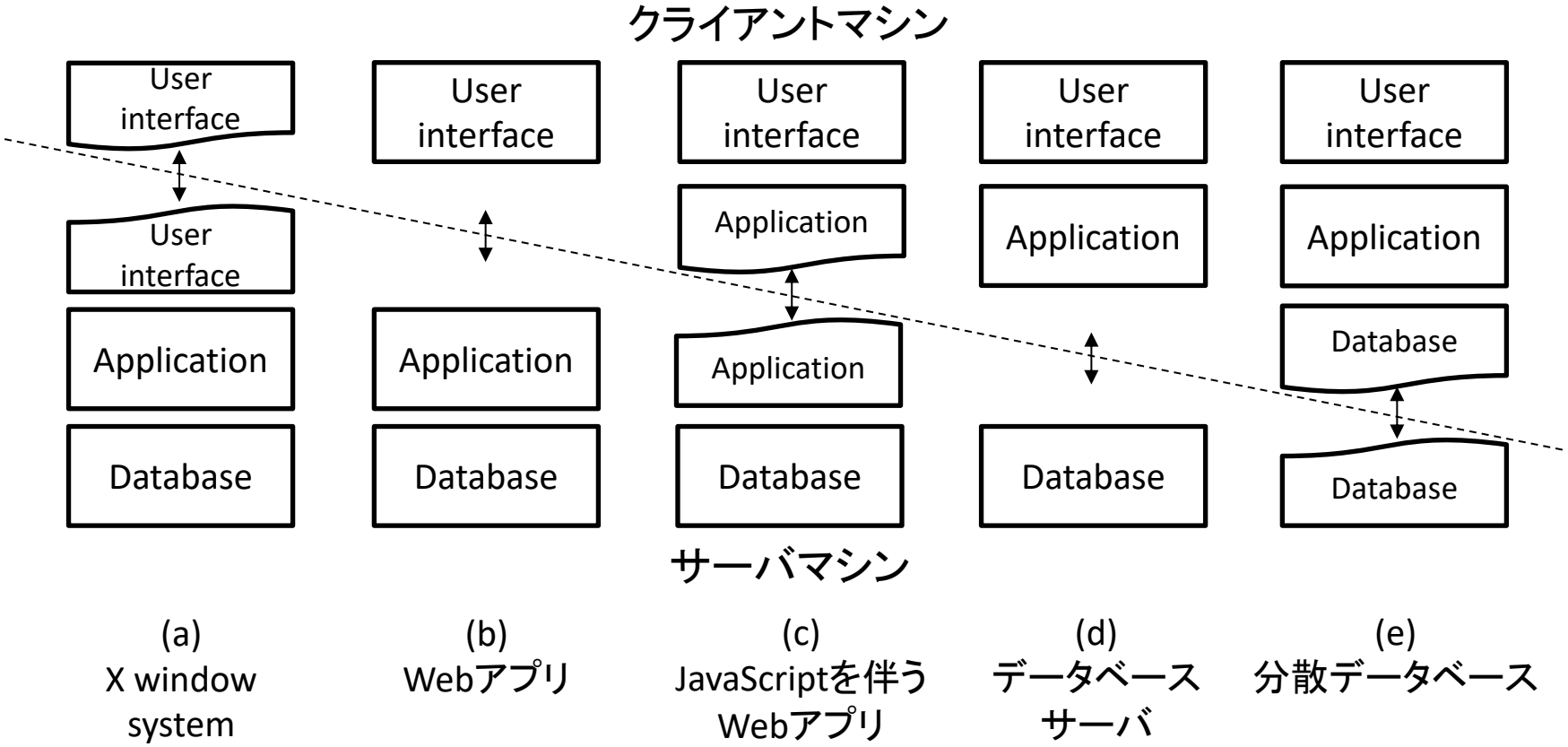


Three-tiered architecture

アプリケーションサーバは，ユーザインタフェースに対してはサーバとして，データベースサーバに対してはクライアントとして振る舞う

実装におけるクライアントサーバの分離

- クライアントサーバシステムにおける物理的な分散形態
- 「物理的2層アーキテクチャ」（**physically two-tiered architecture**）と呼ばれる



2層アーキテクチャの構成

(a) ユーザインタフェースの端末依存部分のみをクライアントマシンに持たせ，アプリケーションがデータ表示を遠隔制御

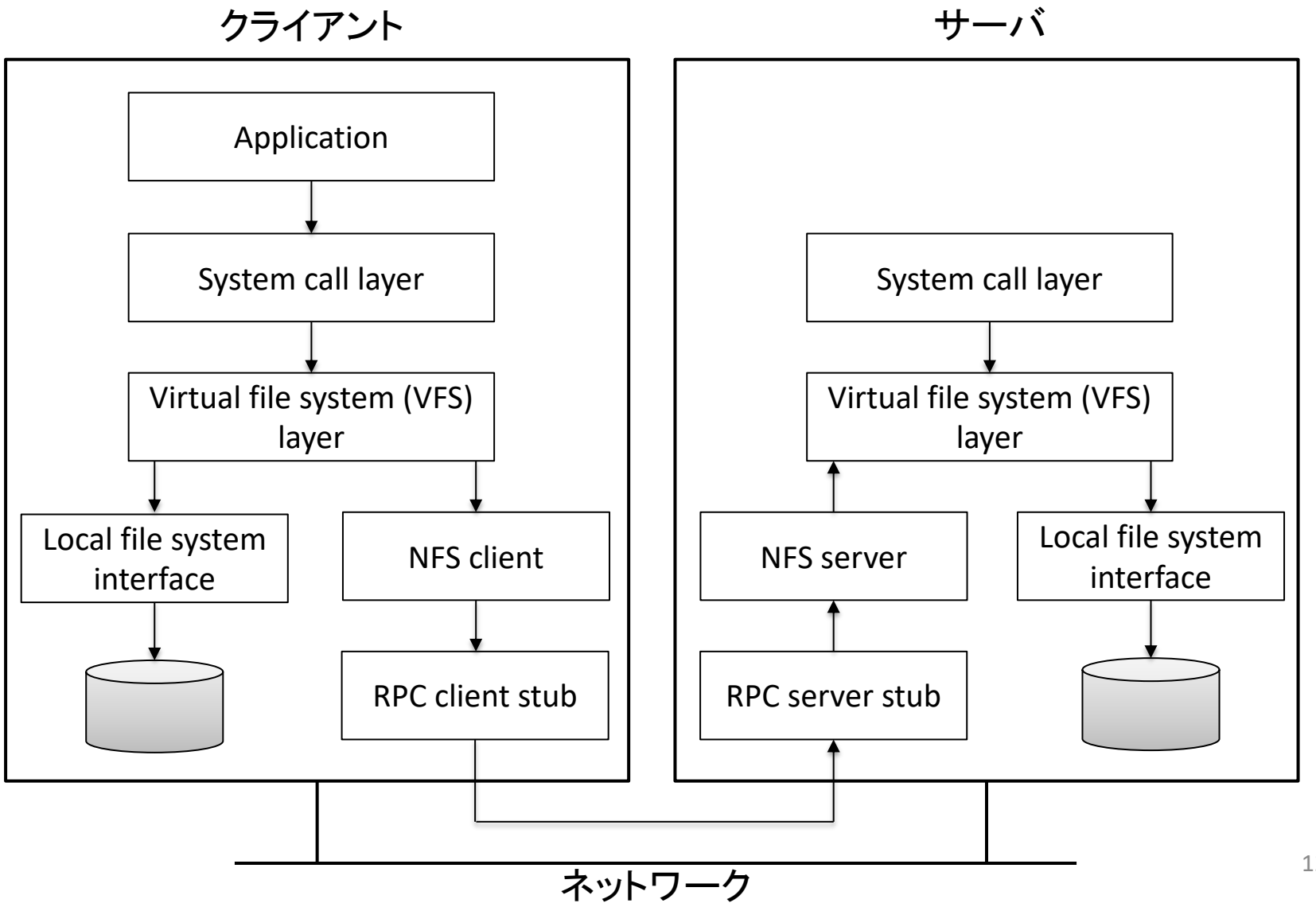
(b) すべてのユーザインタフェースソフトウェアをクライアントに配置（アプリケーション固有のプロトコルで通信）

(c) アプリケーションの一部をフロントエンドに移動
（クライアントマシンで入力値のチェックを行うなど）

(d) ネットワークを介してデータベースにアクセス

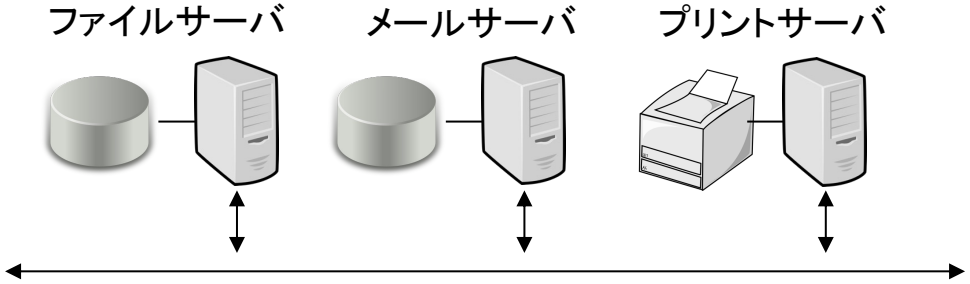
(e) クライアントがデータの一部を保持

The Network File System (NFS)

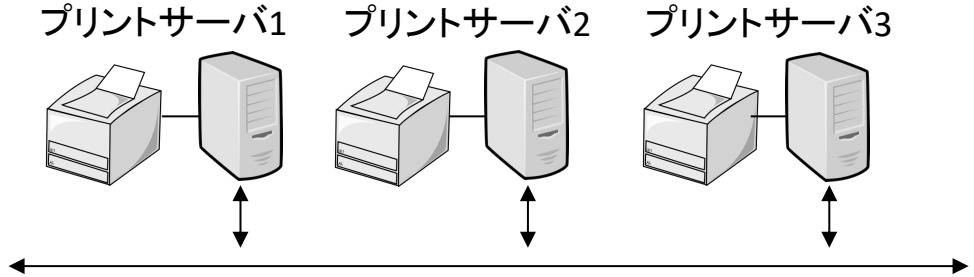


非集中型アーキテクチャ(1)

- 垂直分散
  - クライアントとサーバを異なるコンピュータに分散
  - 各コンピュータの役割に合わせて仕様を最適化



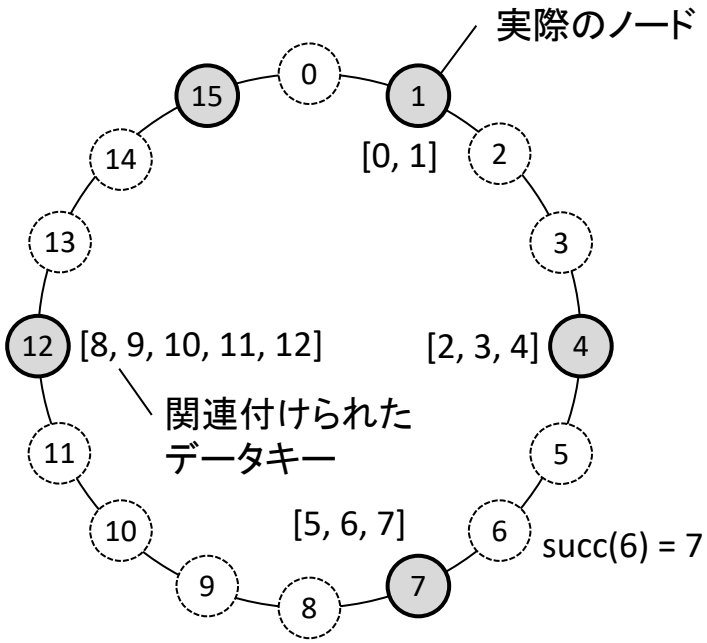
- 水平分散
  - 複数のコンピュータに同種の処理を分散し，負荷を低減
  - 高いスケーラビリティ，耐故障性を実現可能



非集中型アーキテクチャ(2)

- ピアツーピアシステム(peer-to-peerシステム)
  - 同一の複数のプロセスによって構成
  - 各プロセスはサーバでもありクライアントでもある(servent)
  - 実際のネットワーク上にプロセス間を接続する仮想的なネットワークを構成(オーバレイネットワーク)
- 構造化ピアツーピアシステム
  - 規則性をもってオーバレイネットワークを構成
  - 分散ハッシュテーブルのように規則的にノード探索が可能
- 非構造化ピアツーピアシステム
  - 規則性を持たずに(ランダムに)オーバレイネットワークを構成
  - ノード探索には，すべてのノードにメッセージを行き渡らせる「フラッディング」(flooding)が必要

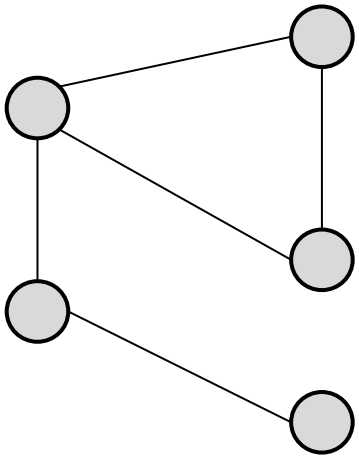
構造化ピアツーピアアーキテクチャ



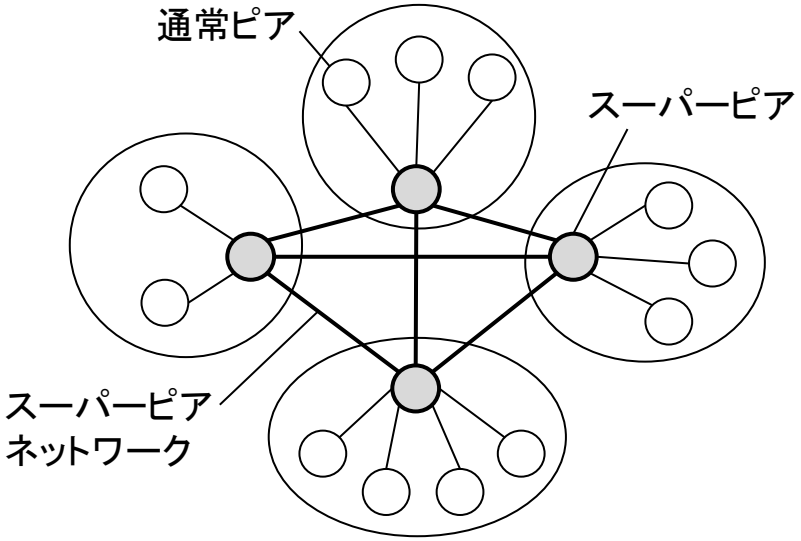
Chordにおけるノード上にある
データ項目のマッピング

- mビットの識別子でリングを構成
- 2^m個の識別子の中からランダムに実ノードを割り当て
- キーkを持つエンティティの情報はid≧kとなる最小の識別子idを持つ実ノード（succ(k)）に格納
- 新しいノードが参加する場合
  - 無作為にidを生成
  - succ(id)のアドレスを探索
  - succ(id)とそのプレデセッサ（predecessor）に連絡し，リング内に挿入
- ノードが離脱する場合
  - succ(id)とそのプレデセッサに連絡

非構造化ピアツーピアアーキテクチャ

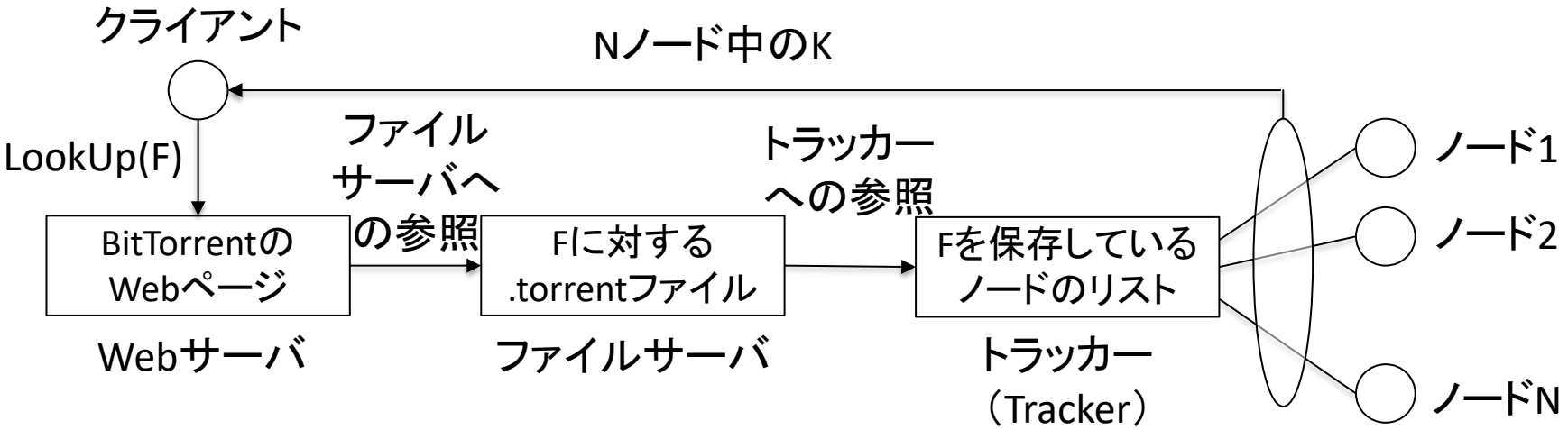


- ランダムグラフに似たオーバレイネットワークを構築
- 各ノードは近隣ノードのリストを保持
- データ項目はノード上にランダムに配置
- 特定のデータ項目の位置を確認するとき，検索クエリをフラッディングしなければならない



- 非構造化ピアツーピアシステムのスケーラビリティを改善
- スーパーピアと通常ピアは階層的に構成
- スーパーピア間はピアツーピアネットワークで接続
- 通常ピア間の通信はスーパーピアを介して行う

BitTorrentファイル共有システム



- トラッカーは要求されたファイルの一部分を持つアクティブノードのアカウントを保持
- アクティブノードは現在別のファイルをダウンロードしているノード
- アクティブノードは，他のマシンへファイルの一部分を提供
- もしノードPがノードQがアップロードしている以上にダウンロードしていることを通知すると，PはQへデータを送信する率を減らすように決める

Pouwelse, Johan A., et al. "A measurement study of the BitTorrent peer-to-peer file-sharing system." (2004).