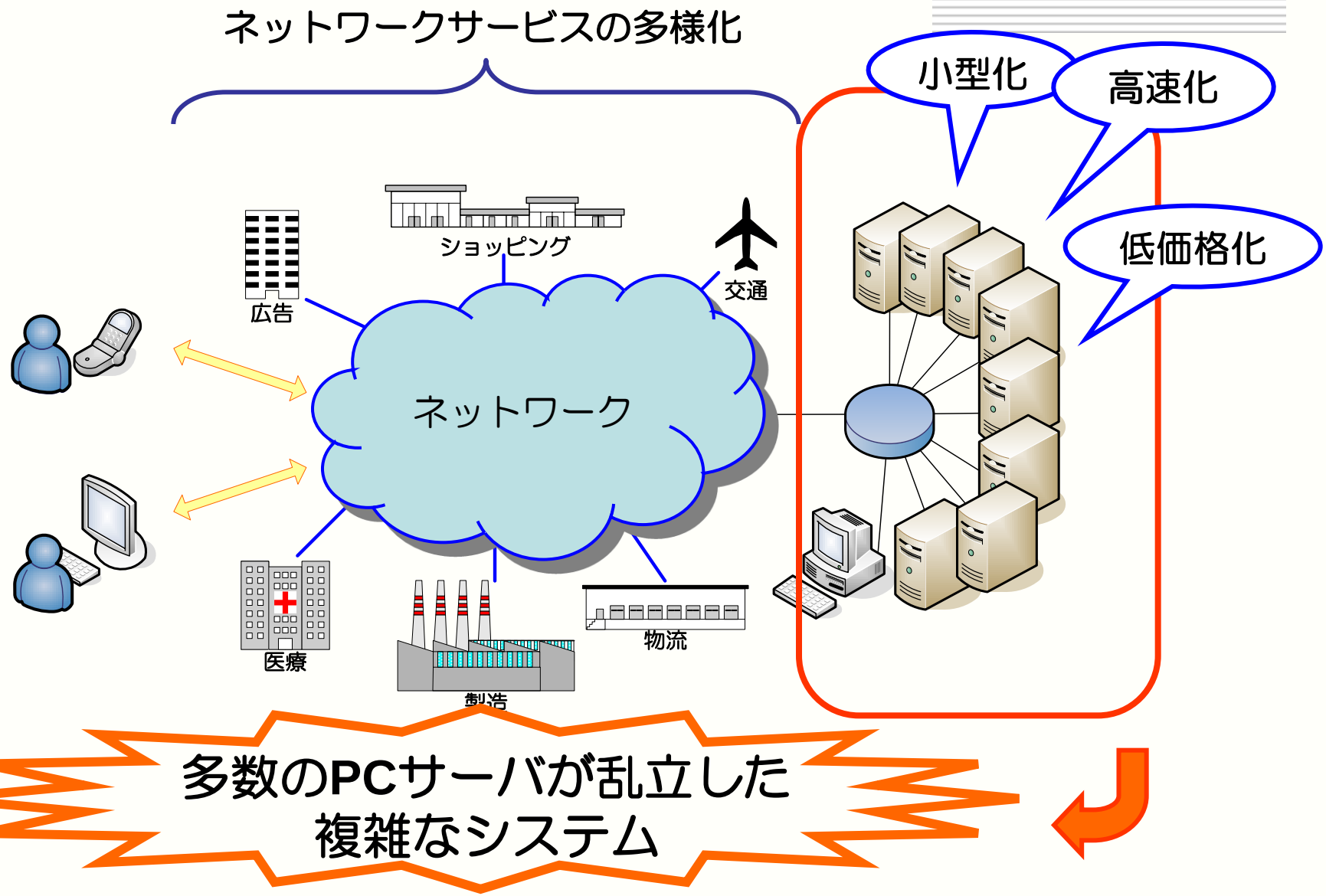
仮想マシン間の同期による高可用クラスタリング方式

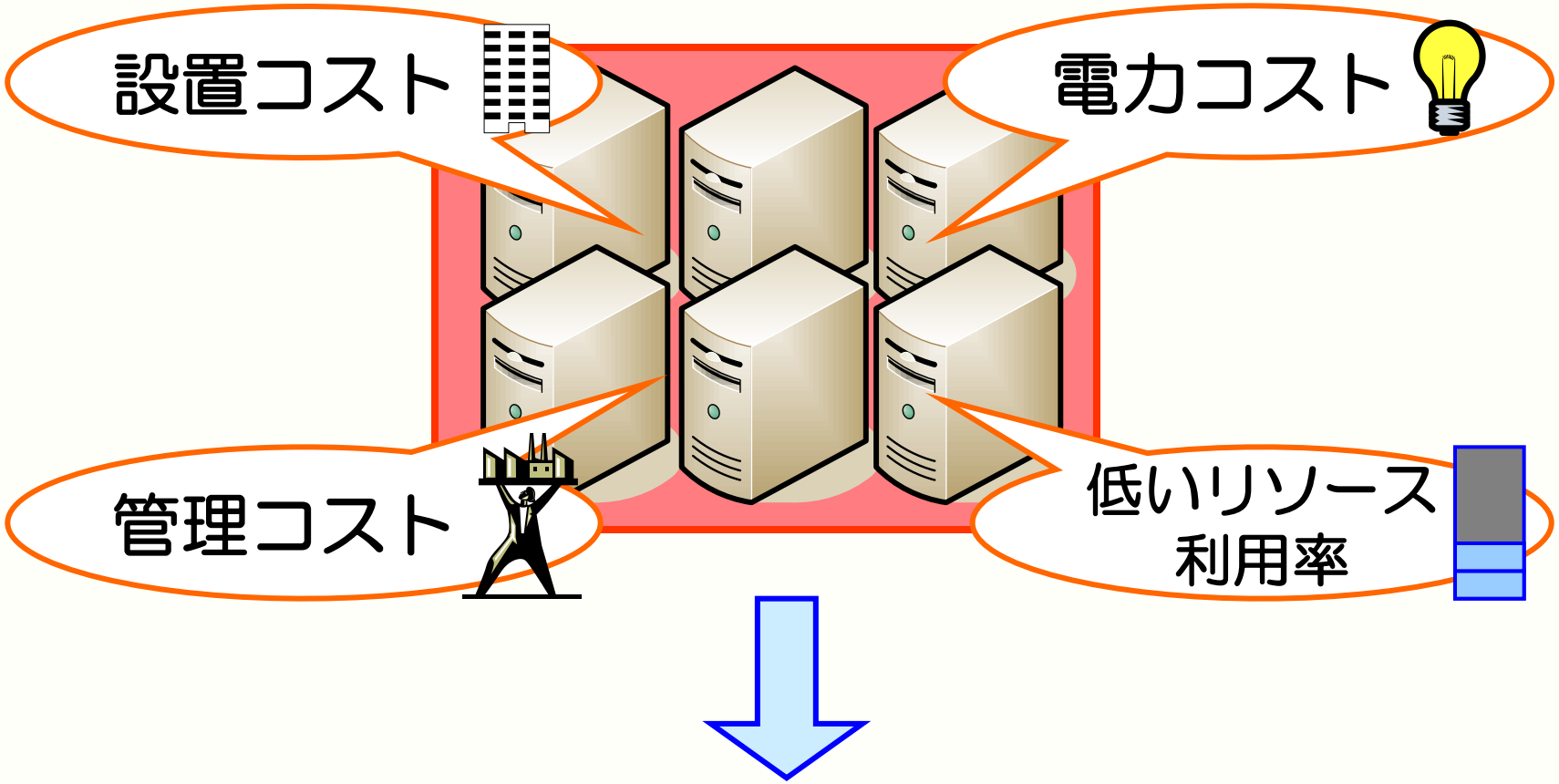
NTTサイバースペース研究所
OSSコンピューティングプロジェクト
カーネルグループ
田村 芳明

2007/7/20

研究の背景



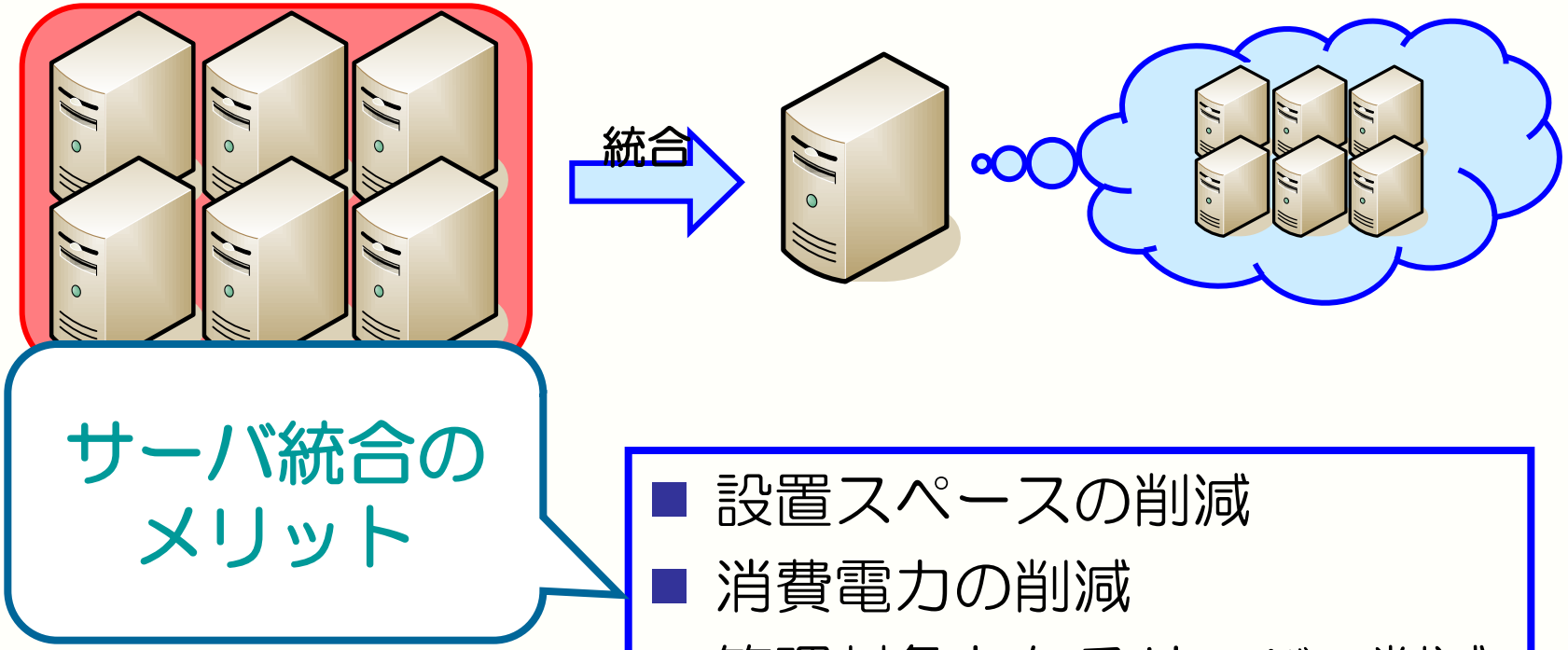
多数のPCサーバによる複雑なシステムの課題



コスト削減とリソースの有効利用が必要

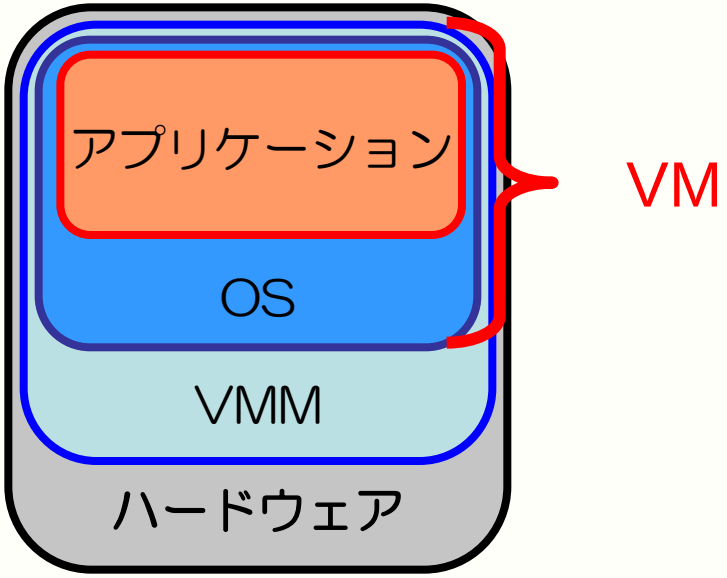
仮想マシンを利用したサーバ統合

- 複数のサーバを1つの物理マシンに統合
  - 複数の仮想マシンが物理マシン上で動作



仮想マシン(VM)とは？

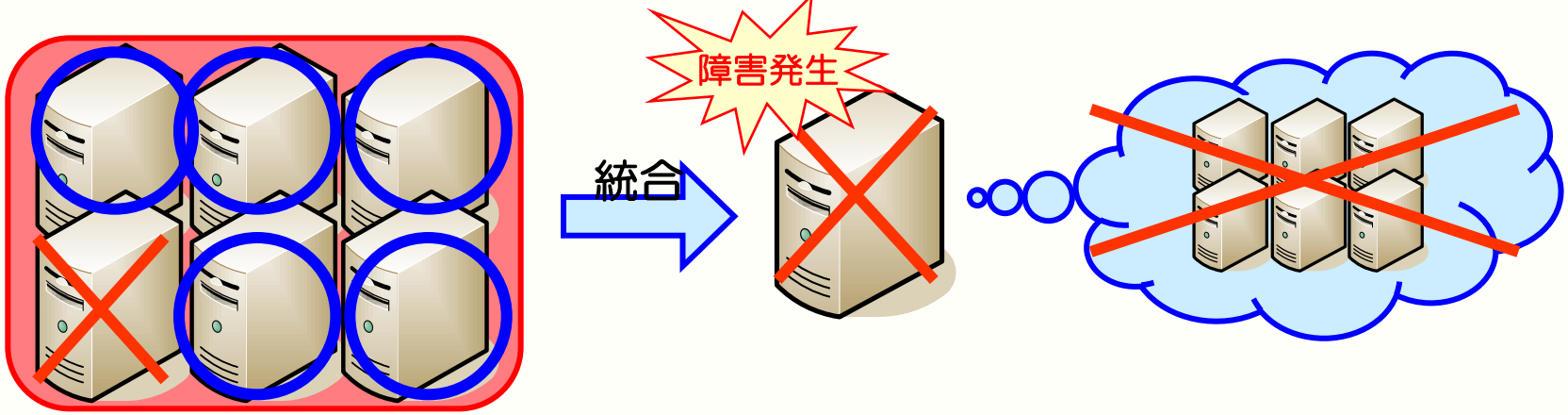
- ハードウェアを仮想化
  - ハードウェアを
    占有しているように見せる
  - 1960年代からある技術
  - 複数のVMを実行可能
- 仮想マシンモニタ(VMM)
  - 物理資源の分割・割り当て
- 有力な製品
  - Xen，VMware



- Intel Virtualization Technology (VT)
  AMD Virtualization (AMD-V)
  - 一般的なx86プロセッサにも仮想化を支援する機能が搭載
  - VM上でOSをより安全・高速に実行できる
- VMの普及が予想される

サーバ統合における課題

- ただ統合しただけでは…



- ハードウェア障害が発生
  → 統合前よりも多くのサービスが停止

サーバ統合を実現するには

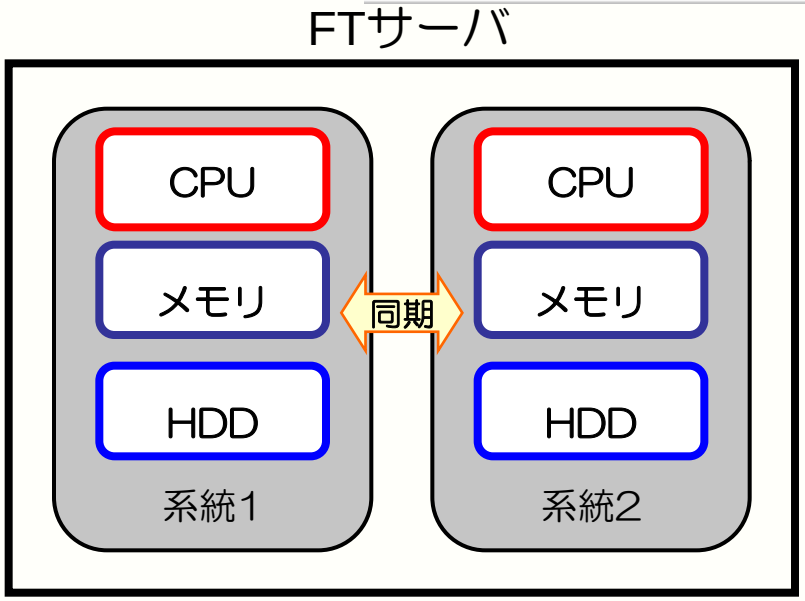
可用性の高い構成が必要

既存の高可用化技術

1. フォールトトレラントサーバ
2. 高可用クラスタ(HAクラスタ)
   - アクティブスタンバイ
   - レプリケーション

フォールトトレラントサーバ

- 特徴
  - 主要なハードウェアコンポーネントを冗長化
    - CPU，メモリ，HDD，バス，電源
  - 一系統に障害が発生した場合，別系統が**透過的**にサービスを継続



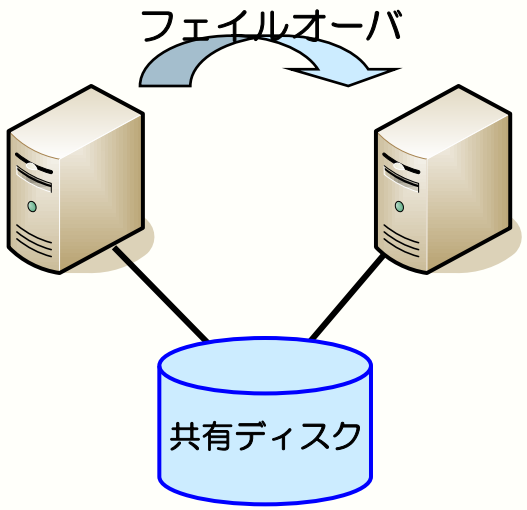
サービスを継続するために，
アプリケーションやOSが特別な処理を必要としない

- 問題

特殊なハードウェアのため高価であり
サーバ統合によるコスト削減に繋がらない

HAクラスタ(アクティブスタンバイ)

- 特徴
  - 運用系と待機系でディスクを共有
  - 運用系が停止した場合,
    待機系にフェイルオーバ
  - フォールトトレラントサーバと異なり,
    安価なハードウェアで構築可能

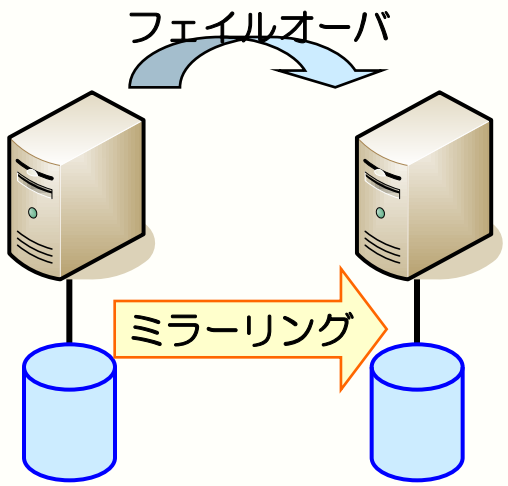


- 問題

透過的に可用性が得られないため
フェイルオーバ時にサービスが中断する

HAクラスタ(レプリケーション)

- 特徴
  - 運用系と待機系が個別にディスクを持つ
  - レプリケーションに
    必要な情報をミラーリング
  - 運用系が停止した場合,
    待機系にフェイルオーバ



- 問題

全てのアプリケーションに対して
レプリケーションの仕組みを作るのは大変

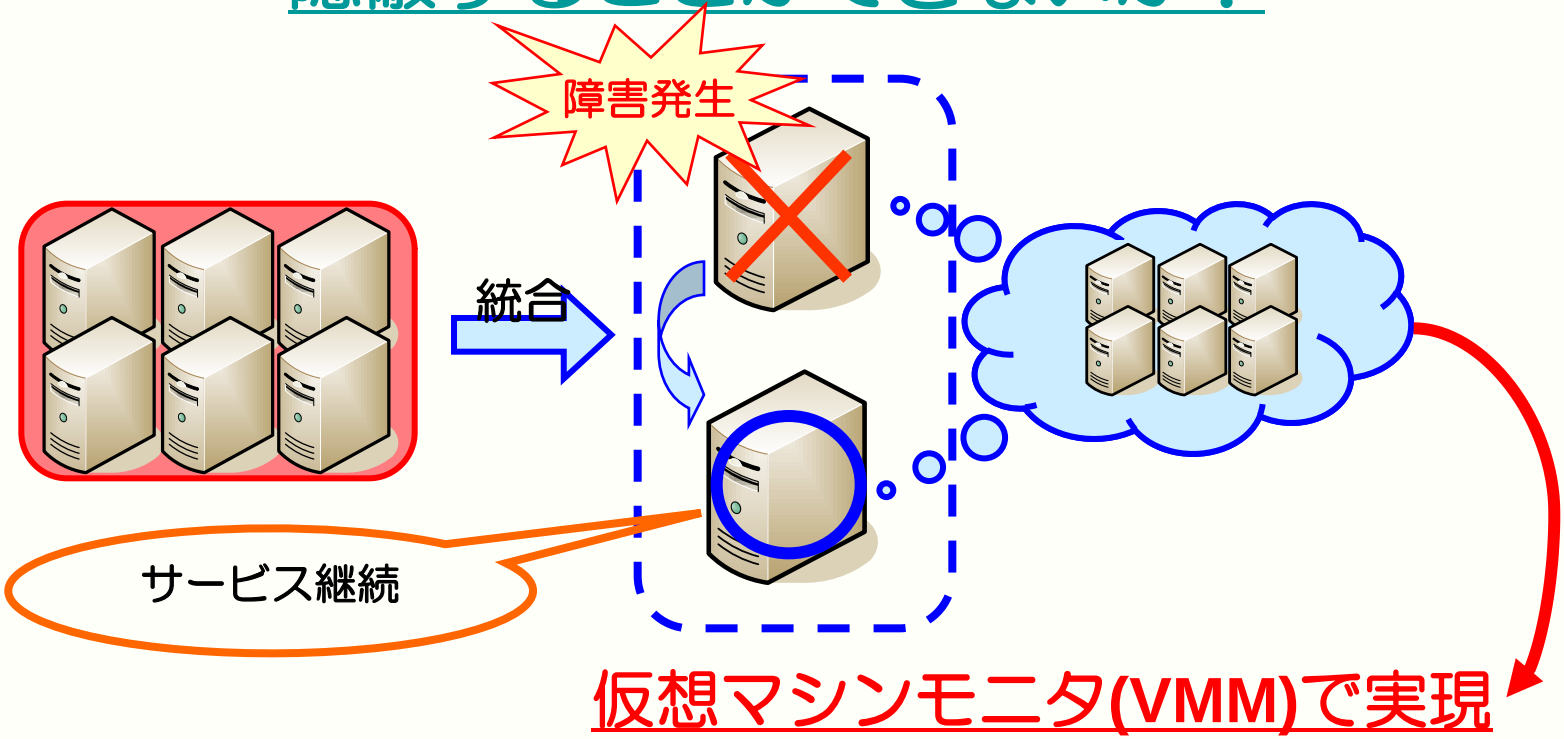
既存の高可用化技術と問題点

既存技術の比較

	透過性	切替時間	コスト
FTサーバ	○ アプリケーションやOSの対処不要	○ ハードウェアで高速に切替	× 特殊なハードウェアのため高価
HAクラスタ(アクティブスタンバイ方式)	× アプリケーションで対処が必要	× 待機系はリカバリが必要なためサービスが中断	○ 汎用的なサーバで構築可能
HAクラスタ(レプリケーション方式)	× アプリケーションで対処が必要	○ 待機系は運用系を切り離して継続	× 全てのアプリケーションにレプリケーションの仕組みを作るのは困難

VMMによるクラスタリング

VMMによって障害発生を隠蔽することができないか？



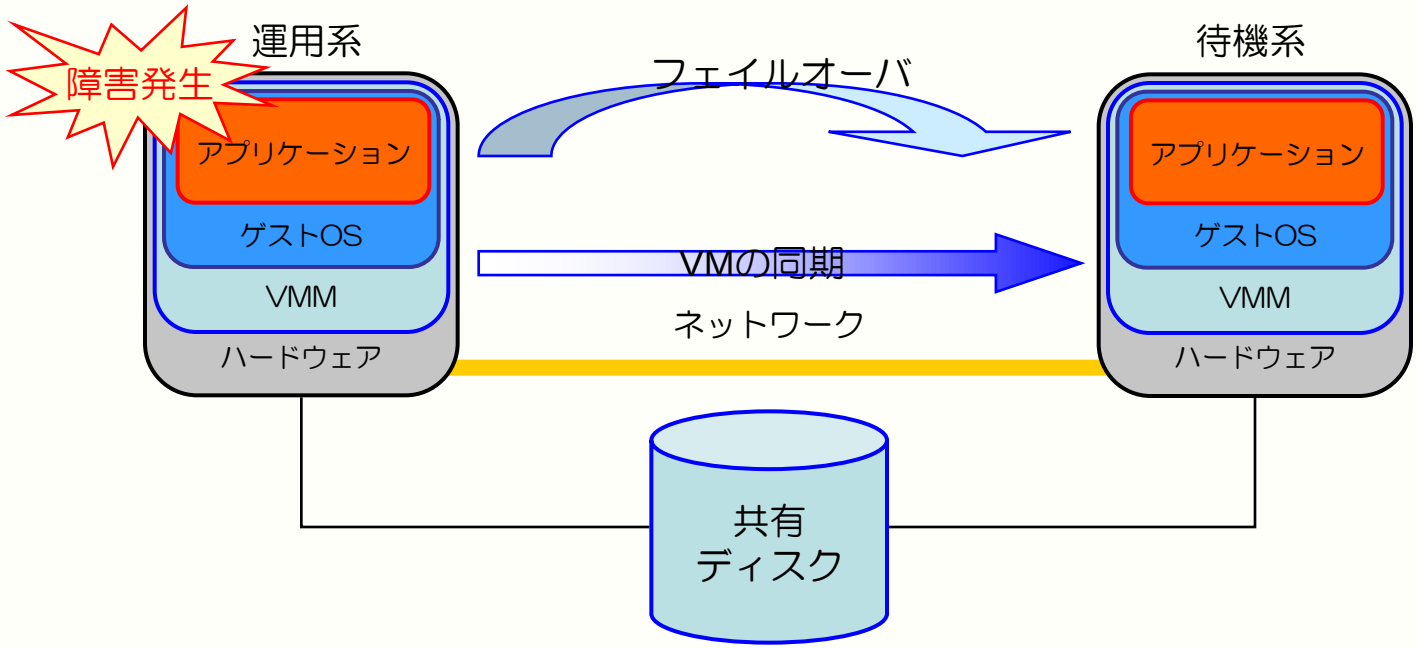
仮想マシンモニタ(VMM)で実現

透過性	切替時間	コスト
○	○	○

VMMによるクラスタリングを実現するには？

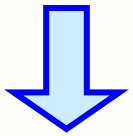
- VMの同期
  - 運用系と待機系で同一の演算結果が得られる
- 障害検知
- 障害発生後の切り替え

既存技術が利用可能



VMの同期を実現するには？

- もっとも単純な方法
  - 1命令を実行するごとに同期
  - FTサーバはハードウェアで実現

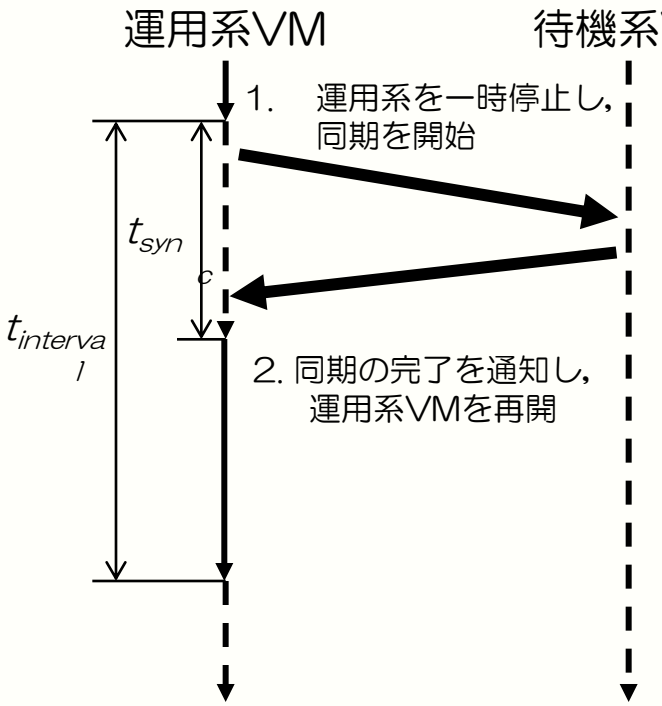


VMMではオーバヘッドが大きすぎる

- 検討課題

VM間の一貫性を
低オーバヘッドで保つ同期方式の実現

仮想マシン間の同期における課題



- 同期のオーバヘッドを小さくする必要がある
  - 同期にかかる時間を短くする
    - ⇒ VMの差分転送が有効
  - 同期の頻度を最小限にする
    - ⇒ 障害発生時にサービスを継続できなければならない

⇓

- VMや外部の状態を変化させるイベントに着目
  - 例: ストレージへの読み書き，ネットワークの送受信
    タイマ割り込み，コンソールの入出力など

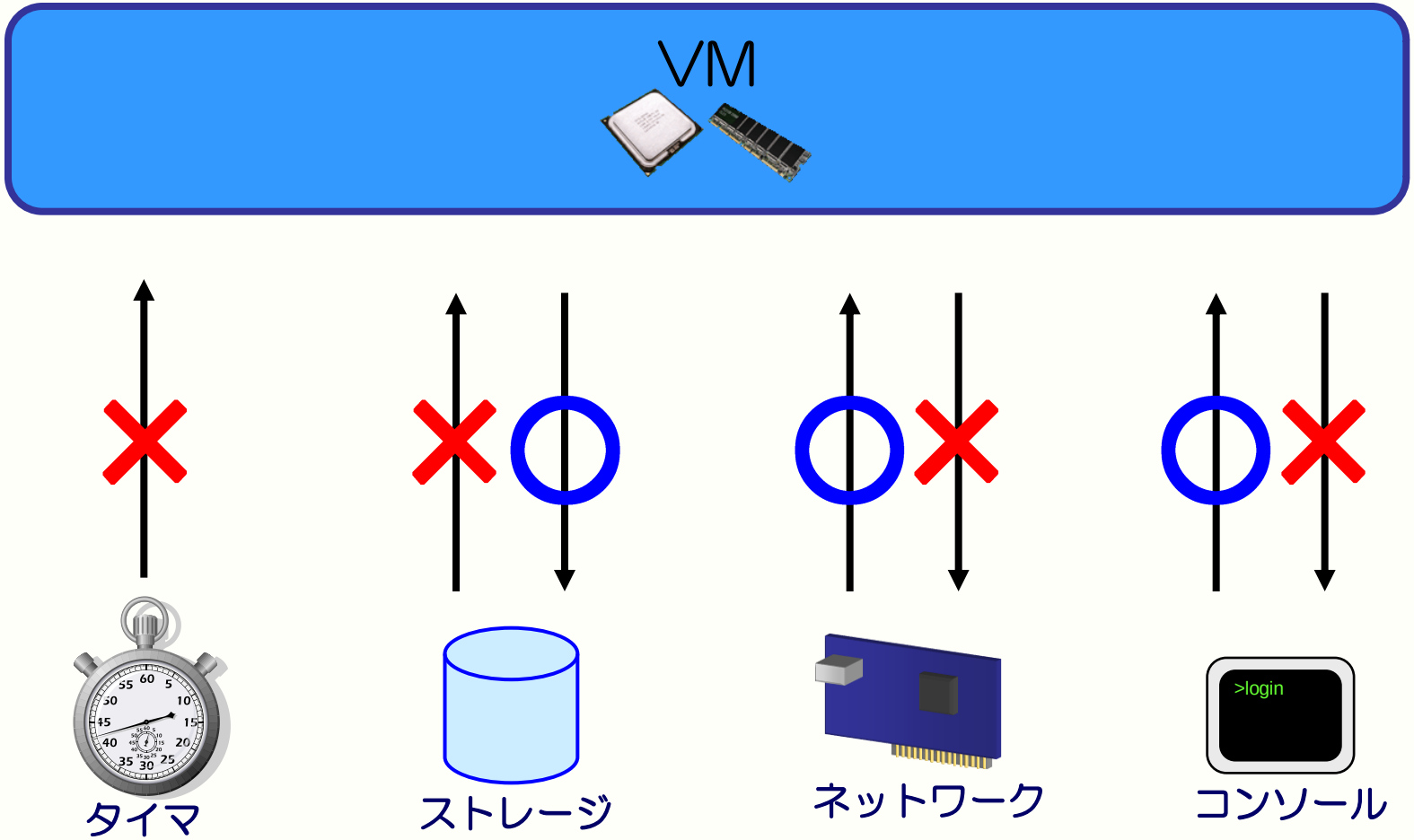
イベントを契機とした同期

- サーバ環境で発生するイベント
  - タイマ，ネットワーク，ストレージが中心
  - ネットワーク，ファイルI/Oの性能に影響
- 仮想化のみの環境と性能を比較（Netperf，IOzoneを使用）
  - ネットワークで24%に低下
  - 同期書き込みで10%，バッファリングで31%に低下

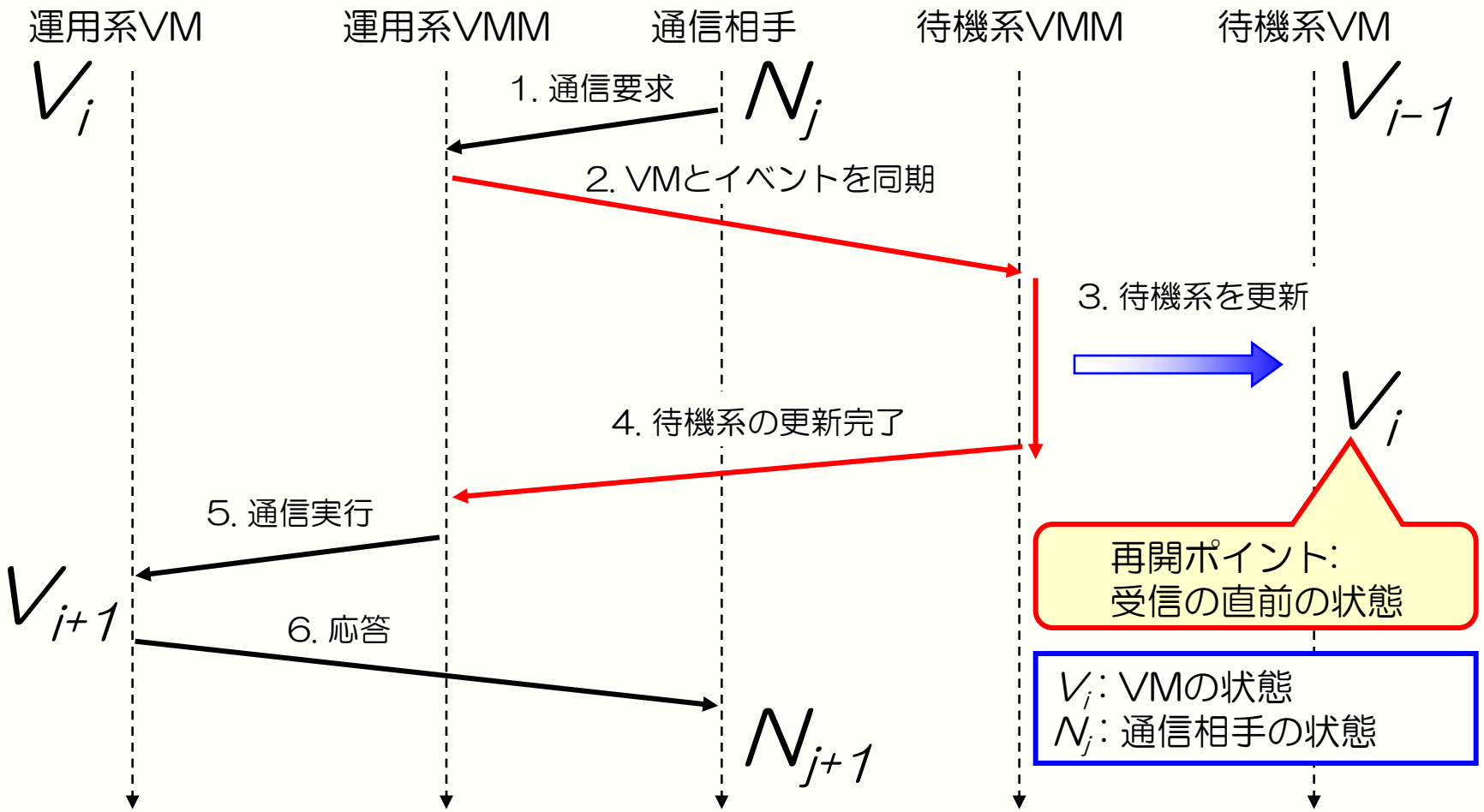
	ネットワーク [Mbps]	ファイルI/O [MB/s]	
		同期書き込み	バッファリング
仮想化のみ	93.39	5.67	53.95
全イベントで同期	22.81	0.58	16.54

- 同期の契機とするイベントの選別が必要
  - 仮想化のみを基準として50%の性能を得ることを目標とする

同期の契機とするイベントの選別

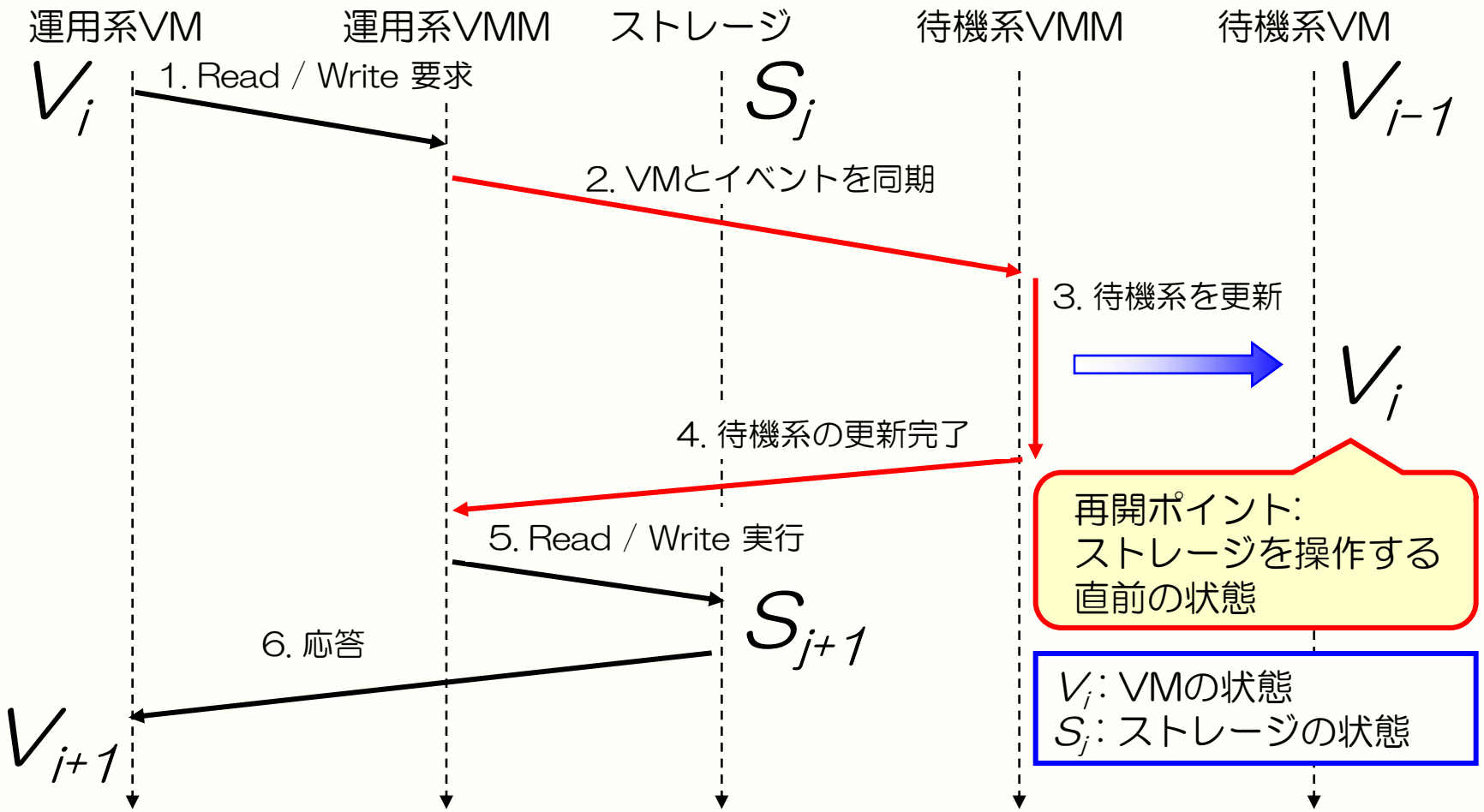


ネットワークからのイベントによる同期



- 運用系と待機系の二重送信によって問題は生じない
  - TCP のような信頼性を保証する通信プロトコルではプロトコルで対処
  - UDPのような信頼性を保証しないものではアプリケーションが対処

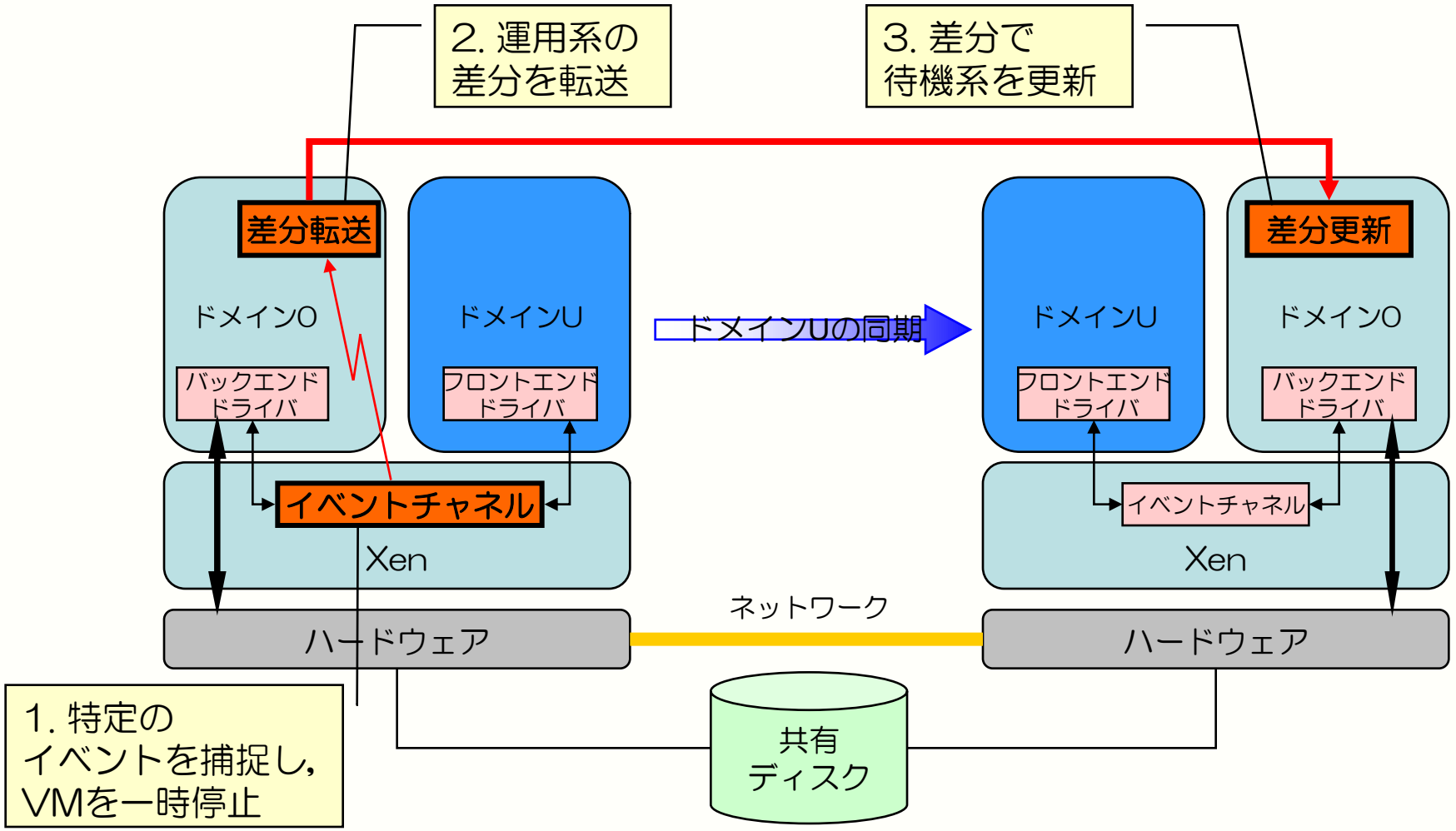
ストレージへのイベントによる同期



- 待機系はストレージに対して，運用系と同一の操作を行うため，ストレージからの応答は再現できる

提案方式の実装

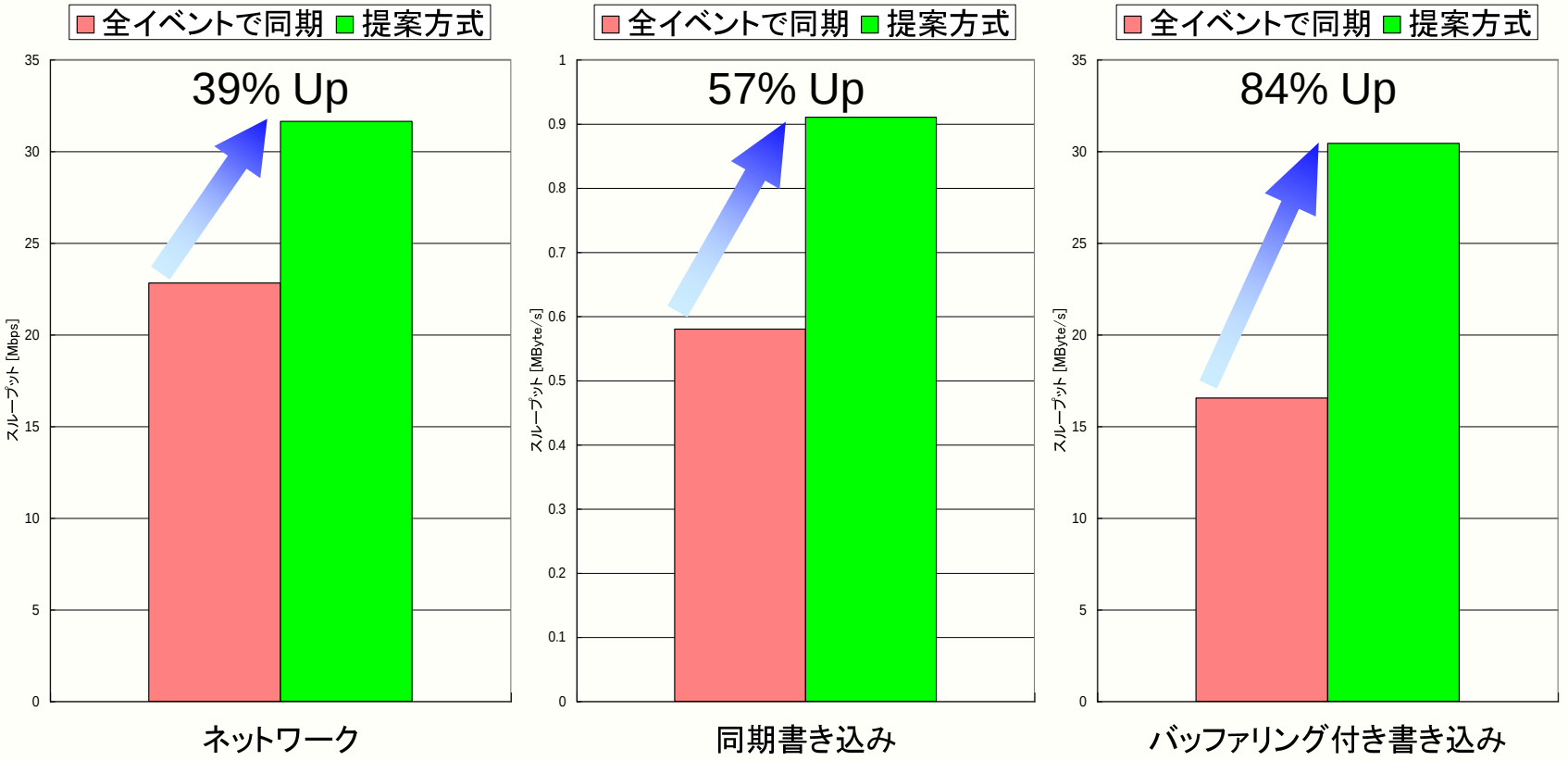
- Xenを使ってプロトタイプを実装



提案方式の評価

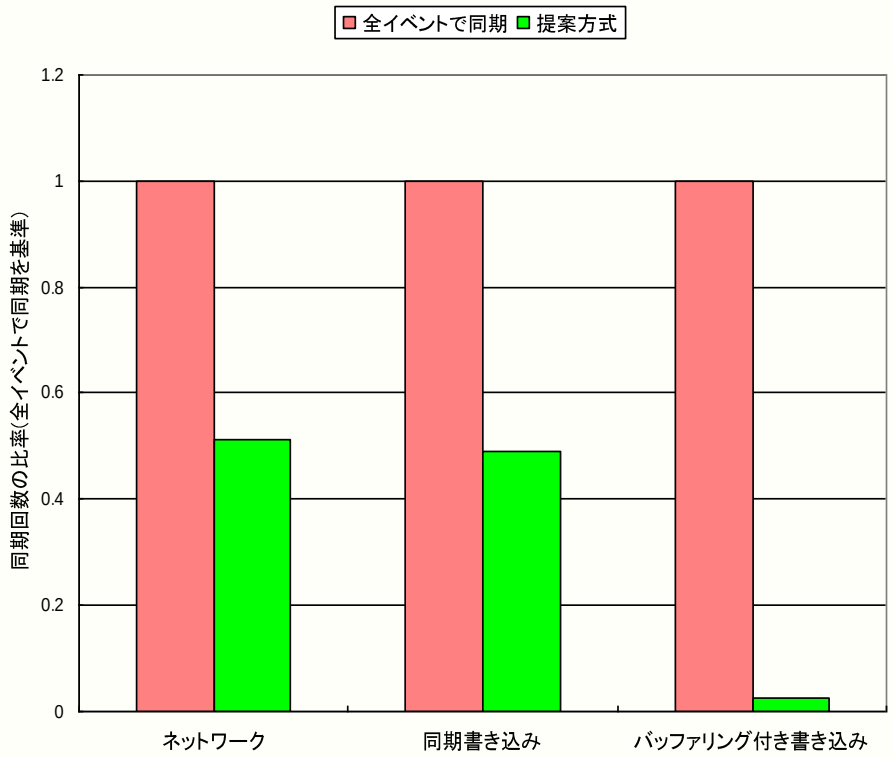
- 評価項目
  - 同期中のネットワーク，ファイルI/Oの性能
  - 同期の回数，ページ転送にかかる時間
- 評価環境
  - サーバ
    - CPU: Intel Xeon 3GHz X 2
    - メモリ: 4GB
    - ネットワーク: Gigabit Ethernet
    - 共有ディスク: FCディスク
  - VM
    - VMM: Xen unstable
    - ゲストOS: Debian Etch Testing
    - メモリ: 1GB

同期中の運用系の性能

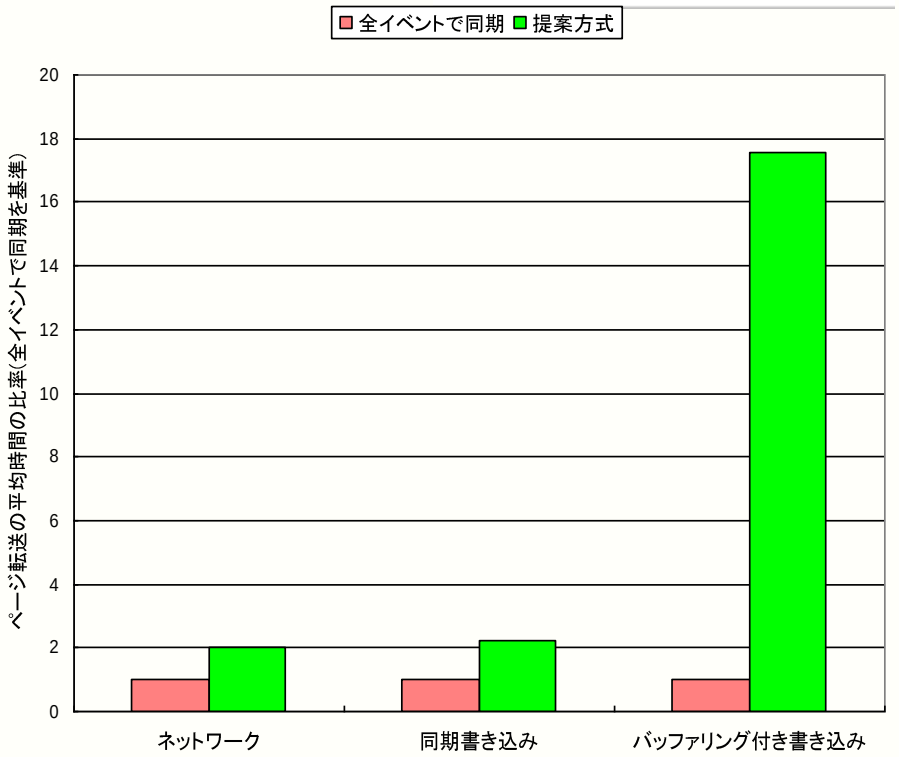


- 同期の頻度を削減したことにより性能が改善

同期回数とページ転送にかかる時間



同期回数



ページ転送にかかる平均時間

- 同期回数は49%～96%削減
- ページ転送にかかる平均時間が2倍～18倍に増加
- 転送処理の最適化，ページの圧縮転送が必要

関連研究

- Hypervisor-based fault tolerance [1]
  - 待機系VMMが運用系をシミュレートすることによって同期
  - PCサーバに実装するのは困難
- SecondSite [2]
  - Xenのマイグレーションを拡張し，VMをストレージに継続的に保存
  - 待機系が処理を引き継げない場合がある
- ExtraVirt [3]
  - プロセッサの故障を対象とし，複数のプロセッサの同期
  - プロセッサ以外の故障には対応できない
- VLS [4]
  - ロギング・リプレイを使い、待機系で不足したページだけを転送する方式
  - Mini-OSは動作済み、Linuxに適用できるように実装中

[1] T. C. Bressoud et al. ACM TOCS '96
[2] B. Cully et al. HotDep '06
[3] D. Lucchetti et al. SOSP '05
[4] D. Stodden et al. Xen Summit '07

まとめ

- VM間の同期によるクラスタリング技術を提案
  - アプリケーションやOS に依存せずに,
    障害発生時にサービスを継続することが可能
  - 低オーバヘッドでVMを同期する方式を提案
- オープンソースのVMMであるXenを利用した
  プロトタイプを実装・評価を実施
- 同期の回数を97%削減するなど,
  提案方式の有効性を確認

今後の課題

- 不足している機能の実装
  - 待機系VMの再開
- 性能改善
  - 転送処理の最適化
- VMMによるクラスタリングの実環境への適用
  - ハードウェア障害が発生した場合の検知方法
  - 障害発生後に待機系に切り替える機能