# データ保護システムの性能評価および
# サイジング方式の研究

田口 雄一

電気通信大学 大学院情報システム学研究科

博士（工学）の学位申請論文

2018 年 9 月

# データ保護システムの性能評価および
# サイジング方式の研究

博士論文審査委員会

| | | |
|---|---|---|
| 主査 | 吉永 努 | 教授 |
| 委員 | 大森 匡 | 教授 |
| 委員 | 田野 俊一 | 教授 |
| 委員 | 大坐畠 智 | 准教授 |
| 委員 | 策力木格 | 准教授 |

# Assessment and Simulation of Data-Protection System Performance

Yuichi Taguchi

## Abstract

The amount of data generated in the world is explosively increasing due to the “digital transformation.” The enterprise companies are trying to acquire valuable knowledge through the data analysis, or to automate business operations using models learned from enormous amounts of data. With the progress of these technologies, the data itself is recognized as a valuable asset today.

Enterprise companies are working on data-protection by various means in order to avoid missing business opportunities due to problem of data access. Disaster recovery is a measure to replicate data to a geographically separated datacenter in order to continue business operations with replicated data even in a case of large-scale disaster. Since the large-scale earthquake occured in Japan, the interest to data-protection is further increasing in enterprise companies.

One of the performance indices of data-protection system is RPO (Recovery Point Objective). RPO is the objective of “recovery point” which is the difference between the time at which the problem occurred to data and the time at which recoverable data is generated. The shorter PRO is the higher performance. For example, if RPO is 60 minutes, the system is required to be able to recover data that is generated within 60 minutes before the time of incident. In other words, this means that data loss within 60 minutes is accepted by the organization policy. In order to control the risk of data loss, enterprise companies must keep running data-protection to achieve RPO.

Therefore, in the management of data-protection system, it is necessary to have a function to monitor a recovery point constantly in order to judge whether it has achieved RPO or not. In order

to monitor recovery point, it is required to measure the transfer delay time from the site where data is created to the site where data is replicated. This research proposes a method to monitor a recovery point. A proposed method defines an abstract model of data-protection system. According to the model, it is possible to calculate recovery point from the input of the amount of write data and the amount of untransferred data that is temporarily stored in upload buffer.

This method makes it possible to judge whether the system achieves RPO or not. If RPO has not been achieved, it is required to enhance system performance with additional system resources. On the other hand, if it is found that the performance is too high, there is a possibility that the system cost can be saved by reducing the amount of resources. So, it is a challenge of this research to discover a best system configuration that achieves RPO and minimize cost as much as possible.

This research proposes a performance simulation process that is used to estimate an appropriate system size. In this method, a calculation formula to estimate the amount of untransferred data stored in the upload buffer is defined. With this estimation, it becomes possible to calculate a recovery point and to find an appropriate system configuration.

These proposed methods are verified through experiments. As for performance evaluation, it is possible to calculate a recovery point from the input of untransferred data amount. Also, in the verification of the simulation of untransferred data amount, we can reduce the gap of simulation value and measured value up to 1.9%. This error does not make gaps after the conversion to the time of recovery point. So I have concluded that it is sufficiently accurate as a simulation. By applying this simulation to system sizing, there is a possibility to reduce 89% of system cost in comparison to the configuration that is designed to fit a peak workload. Through these experiments, the effectiveness of this proposal is demonstrated.

# データ保護システムの性能評価および
# サイジング方式の研究

田口 雄一

## 概　要

近年，デジタル化の進行によりデータ量が爆発的に増加している。企業や政府では，膨大な量のデータ分析による新たな知見の獲得や，データから学習したモデルによって様々な業務プロセスを自動化するといった試みが盛んに進められている。こうした技術の発展により，データそのものが価値ある資産と認知されるようになった。

企業はデータ消失を原因として事業機会を逸することを避けるため，様々な手段によるデータ保護に取り組んでいる。そのひとつであるディザスタリカバリは，地理的に離れた拠点にデータを複製することで，大規模な障害が発生しても，複製されたデータで事業を継続しようとする対策である。東日本大震災以降，企業では遠隔サイトにデータを冗長化する「データ保護システム」への関心がさらに高まっている。

データ保護システムの性能指標のひとつが RPO（Recovery Point Objective）である。RPO は，データに問題が発生した時刻と，同時刻から遡って復旧可能である時刻との差である「リカバリポイント」の目標値であり，短時間であるほど高い性能が要求される。一例として RPO が 60 分に設定されていれば，同システムは問題発生時刻から遡って 60 分以内のデータに復旧できることを求められる。言い換えれば，これは 60 分未満のデータ消失を許容することを意味している。データ消失リスクをコントロールするために，企業は RPO を達成するよう適切にシステムを運用しなければならない。

そのためデータ保護システムの運用管理においては，稼働しているシステムのリカバリポイントが RPO を達成しているかどうか，常時監視する仕組みが必要となる。リカバリポイントを監視するためには，データの生成時刻と，同データが別拠点に記録され

る時刻の差（転送遅延時間）を測定すれば良い。しかしながら，この時差を測定する汎用的な手段が存在しないため，RPO を達成しているかどうか判定できないことが課題となる。そこで本研究では，データ保護システムの構成と処理手順を抽象モデルで表し，リカバリポイントを計算によって予測する汎用的な性能評価方式を提案する。この方式では，転送処理の過程でバッファに滞留するデータ量を入力として遅延時間を計算することにより，リカバリポイントの監視を可能とする。

リカバリポイントの監視が可能になることで，同システムが性能目標 RPO を達成しているかどうか判定できるようになる。判定の結果，RPO を達成していなければシステムリソースを増強して性能を上げる必要がある。逆にもし性能が過剰であると判定された場合には，リソースを削減することでその費用を削減できる可能性がある。その実現にあたっては，RPO を達成し，かつできるだけ費用が少なくなるシステム構成を発見することを目指す。

本研究では，上記の条件を満たす適正なシステムサイズを予測する性能シミュレーション技術を提案する。この方式では，バッファに滞留する未転送データ量を算出する計算式を定義する。この予測値をさらにリカバリポイントに換算することで，PRO を達成する構成であるかどうか判定可能となる。同シミュレーションの適用により，性能目標を達成し，かつリソース量が過剰にならない適切なシステム構成を導出することができるようになる。

以上のシステム性能評価およびサイジング方式を提案し，実験を通じて検証した。性能評価については，実験環境で観測された未転送データ量を入力として，リカバリポイントを計算できることを示した。システムサイジング方式の検証にあたっては，シミュレーションによる未転送データ量の予測値と，実測による測定値の誤差を 1.9%まで縮めることができた。これはリカバリポイントに換算すると誤差が発生しない程度の違いでしかなく，シミュレーションとしては十分に高精度であるとの結論に至った。またこのサイジング方式の適用により，負荷のピークにあわせた構成と比較して 89%のシステムコストを削減できる可能性があることがわかった。これらの実験と考察を通じて，本提案の有効性を実証した。

本研究により，データ保護システムの運用管理 PDCA サイクルを確立し，RPO（すなわちデータ消失リスク）に応じたシステム性能の調整を可能とした。特に通信回線をは

じめとするシステムリソース量を，リスクとコストに従って適切に決定することが可能となった。

# 目次

# 図目次

# 表目次

# 1章　序論

## 1.1. 背景と目的

今日，情報システムは企業活動にとって不可欠な基盤である。企業情報システムは日々大量のデータを生成し，その蓄積量は増加の一途をたどっている [1] [2] [3]。さらに企業が保有する膨大なデータから新たな知識や情報を発見しようとする試みが多くなされているように，データそのものが価値ある資産と認識されるようになっている [4] [5] [6] [7] [8]。また企業ではひとたびデータ消失が起これば事業機会を逸するだけでなく，顧客からの信頼や社会的信用を失うといった重大なリスクが認知されている [9]。このようにデータ保護は企業経営にとって重要課題のひとつであり，様々な対策が講じられている [10] [11] [12] [13]。データ保護には重要度に応じていくつかのレベルがある。重要なデータについては，一般的なストレージの冗長化 [14]やローカルバックアップにとどまらず，火災や停電のような拠点規模の損害，さらには地震や水害など自然災害による広域被災への対策が求められる。こうした大規模災害においてもデータを保護し，業務継続を可能とするために，データ保護システムが有用である。データ保護システムは，距離を隔てた二つ以上の拠点にデータを複製し，冗長化しておくことで，ある拠点で障害が発生した状況にあっても，別の拠点からデータを復旧して業務を継続しようとするものである [15] [16]。

このとき，データを生成する拠点から複製データを保管する拠点へのデータ転送が遅延すると，その時点で冗長化されないデータが発生する。冗長化されていない状態のデータは，インシデント発生時に失われるリスクがある。このデータ保護システムにおけるデータ消失リスクは，インシデント発生時刻と，同時刻から遡って復旧可能な時刻との差である「リカバリポイント」に該当する。言い換えると，このリカバリポイントが短時間であるほど，高いデータ保護性能が要求されるということになる。本研究では，リカバリポイントを計算し，常時監視可能とする性能評価方式を提案する。

ディザスタリカバリを目的とする従来のデータ保護システムは，ストレージやデータベースのコピー機能で実装されてきた。これはデータを生成する拠点と，その複製を保管する拠点のそれぞれに対となるストレージあるいはデータベースを設置し，継続的に

データをコピーするものである。これらの機能はストレージやデータベースの開発元ベンダによって設計された独自仕様であり，性能管理をはじめとする運用方式もあわせて同ベンダによって提供された [17]。拠点間の回線帯域をはじめとするシステム構成もまた，同ベンダの協力のもと利用者がほぼすべて設計する必要があった。

その後クラウドが普及すると，複製データを保管する待機用のサイトを企業が自ら設置するのではなく，代わりにクラウドをバックアップデータの保管先とすることで大幅なコスト低減を見込めるようになった [18]。その実装方式のひとつが，クラウド事業者が提供するゲートウェイサービスである。ゲートウェイはデータを生成するユーザの拠点（オンプレミス環境）で動作し，書き込まれたデータを継続的にクラウドに転送する。このクラウドゲートウェイの運用においては，データ転送遅延時間の監視および制御が課題となる。クラウド事業者は従来のストレージ事業者のようにデータ保護性能に責任を負わないため，データ消失リスクの管理機能が提供されない。そこで本研究ではクラウドゲートウェイにも適用できる，標準的なパラメータだけを用いたシステム性能評価方式の確立を図る。

データ消失リスクを低減するためには十分な量のシステムリソースを設けて高性能化すれば良いが，これはコスト増の要因にもなり得る。したがって，システムへの負荷および性能，コストのバランスを定期的に検証し，常に適正なシステムサイズに調整することが求められる。本研究では性能目標を達成し，かつリソース量が過剰でない構成を見積もる性能シミュレーション方式を提案する。性能シミュレーションを通じて負荷に応じた適正システム構成を導出可能することで，システム運用管理における PDCA サイクルを確立し，コスト低減に寄与する。

まとめると，本論文は以下の貢献を果たす研究に位置づけられる。

- データコピー機能の実装に依存せず，標準的なパラメータだけで実現するデータ保護システム性能（リカバリポイント）評価方式の確立
- 性能シミュレーションを通じて，システム性能とコストを適正化するシステム構成設計（システムサイジング）方式の確立

- クラウドの普及によりシステム実装や運用管理手法が変化したデータ保護にあっても，そのシステムリソース量を調整してトータルコストを抑える運用管理サイクル（PDCA サイクル）の確立

## 1.2. 論文の構成

本論文は，全８章の構成である。各章の構成を図 1.1 に示す。

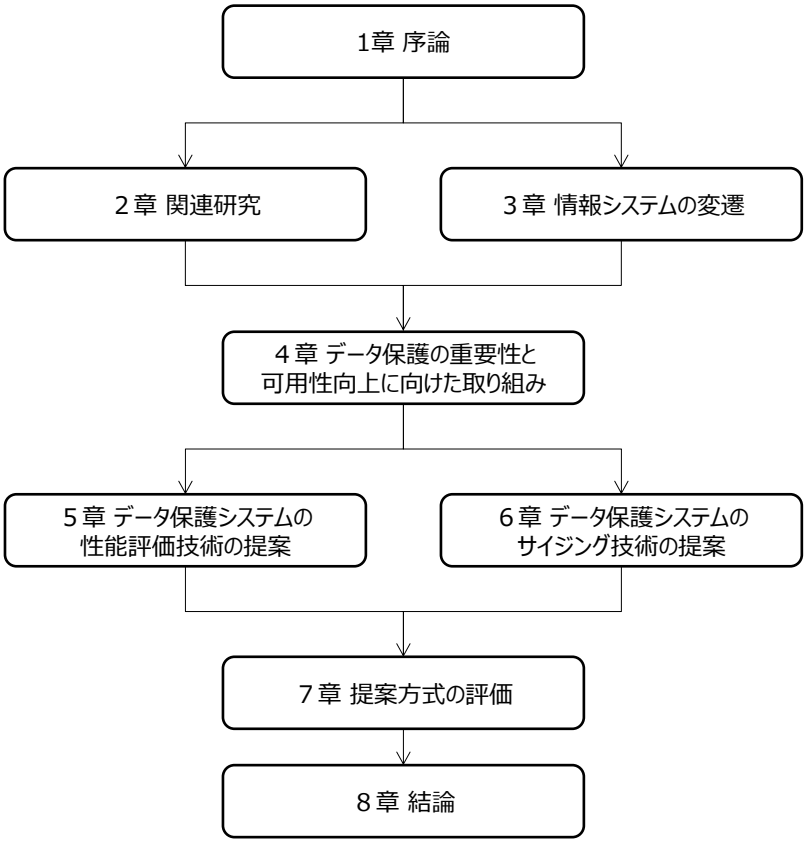



**図 1.1 論文の構成**

第 2 章では，本研究に関連する過去の研究について述べる。ディザスタリカバリの計画やシステム要件，性能監視に関わるこれまでの研究をまとめた上で，本研究の位置づけを明らかにする。

第 3 章では，本研究の対象である情報システムの変遷について述べる。特にアーキテクチャ，システム開発手法および運用管理手法の変化の過程を整理し，本研究の対象領

域を明らかにする。また本研究の背景として，データ保護の重要性がより高まってきた理由のひとつである，データ利活用技術の潮流について述べる。

第 4 章では，データ保護に対する期待の高まりと，データ消失リスクをコントロールすることの必要性について考察する。データ転送遅延によって生じるデータ消失のリスクを定義し，その性能指標であるリカバリポイントの運用管理が本研究の目的であることを示す。さらにデータ保護システムの実装方式を整理し，本論文のターゲットを明確にする。

第 5 章では，データ保護システム性能評価方式を提案する。本方式では，データ保護システムの構成と振る舞いをモデルとして定義し，同モデルにもとづいてリカバリポイントを算出する計算式を策定することで，日常的な監視を可能とする。

第 6 章では，データ保護システムの構成を適正化するサイジング方式を提案する。これは机上設計したシステムの性能を予測するシミュレーション技術である。同シミュレーションを様々な想定で適用することで，性能要件を達成し，かつコストが最小となるケースを選択することが可能となる。

第 7 章では，ストレージを使ったデータ保護システムと，クラウドサービスを使ったシステムのそれぞれについて，上記の性能評価方式およびサイジング方式の検証を行なう。実際に企業データセンタで発生した負荷（書き込みデータ量）を入力としたリカバリポイントの予測を行い，その妥当性を考察する。また，クラウドサービスで実装したデータ保護システムの実験環境を用い，性能シミュレーションの精度を検証する。さらに負荷のピークにあわせたシステム構成と比較して，システムサイジングの適用によるコスト削減効果を見積もり，本研究の有効性を考察する。

# 2章　関連研究

## 2.1. 関連する研究カテゴリ

クラウドコンピューティングは，グローバル規模に分散された IT リソースを自在に共有できる優れた特長により，普及の一途を辿っている [19]。一方で，クラウドにおいてはビジネスの継続性やユーザ満足度を向上するために考慮すべきリスク管理，障害発生後の回復の仕組みなど，セキュリティに関わる様々な問題が提起されている [20]。図 2.1 に示すとおり，自然災害だけでなく操作ミスなど人災によるトラブルは，高価なサービスの中断を招く原因となる。これらのリスクを避けるため，ディザスタリカバリの研究が進められてきた。文献 [20]によれば，データにトラブルを生じた原因のうち，自然災害は 50%に留まり，それ以外は電力供給の停止やサーバのハードウェア障害，セキュリティによる侵害などに起因している。したがって，ディザスタリカバリは希に起こる自然現象だけでなく，停電や操作ミスなど発生頻度の高いあらゆる障害への対策であることが求められる。

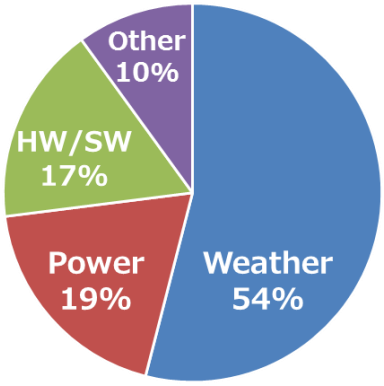



**図 2.1 事業停止の原因**

表 2.1 にディザスタリカバリ，すなわちデータ保護に関わる研究を分類する。このうち，“計画と運用に関する研究”が本研究に関連するカテゴリに該当する。以下これらの関連研究についてまとめた上で，本研究の位置づけを明らかにする。

**表 2.1 関連研究カテゴリ**

| # | 研究カテゴリ | テーマ・内容 | 節 |
|---|---|---|---|
| 1 | 実行に関する研究 | 通常稼働時のデータコピー技術 | － |
| 2 | | 障害発生時のリカバリ技術 | － |
| 3 | 計画と運用に関する研究 | システムコスト | 2.2 |
| 4 | | システム性能設計 | 2.3 |
| 5 | | システム運用管理 | 2.4 |

## 2.2. データ保護システムのコスト

文献 [9]では，システム停止によって生じるロスコストを見積もる簡易な計算式を提案している。

**ダウンタイム１時間あたりの平均コスト**
**＝ １時間あたりの従業員コスト × システム停止の影響を受ける従業員数**
**＋ １時間あたり平均収入 × システム停止の影響を受ける収入**

さらにダウンタイムによるロスコストについて，ひとつの典型例が示されている。

**表 2.2 システムダウンタイム１時間あたりのロスコスト**

| # | 業種 | コスト |
|---|---|---|
| 1 | 証券取引 | $6,450,000 |
| 2 | クレジットカード認証 | $2,600,000 |
| 3 | Ebay | $225,000 |
| 4 | Amazon.com | $180,000 |
| 5 | 宅配サービス | $150,000 |
| 6 | 通信販売 | $113,000 |
| 7 | 航空券予約 | $89,000 |
| 8 | ATM サービス | $14,000 |

表 2.2 のとおり，文献 [9]によれば被災時のロスコストは業種やサービスによって異なる。またデータ保護は企業にとっての重要課題である一方，それ自体が利益を創出するものではないため，コストをできるだけ抑えることが求められる。したがって，データ保護には被災時のロスコストを上回るコストをかけるべきではないと判断できる。

文献 [18]ではデータ保護にかかる一年あたりのシステムコスト $Cost_{Total}$ を下記のように定義している。

$$Cost_{Total} = Cost_{Initial} + Cost_{Ongoing} + Cost_{disaster}$$

$Cost_{Initial}$ は初期導入コストの総額を表す。運用コスト $Cost_{Ongoing}$ は，年間のストレージコスト・データ転送コスト・計算機コストの合計に一致する。$Cost_{disaster}$ はリカバリ費用など被災時に発生するコストの総額とされる。

文献 [21]ではバックアップサイトのサービスレベルを 3 段階に区分し，そのコストを比較している。これらを表 2.3 に示す。#1 のオンサイトはバックアップシステムを別の拠点に設けるのではなく同一サイト内に構築する方式，#2 のコロケーションはデータセンタ事業者から建物や設備を調達する方式，#3 はバックアップにクラウドサービスを利用する方式である。

**表 2.3 バックアップサイトの運用**

| # | 方式 | 同期頻度 | バックアップの独立性（距離） | $Cost_{Initial}$ | $Cost_{Ongoing}$ | $Cost_{disaster}$ |
|---|---|---|---|---|---|---|
| 1 | オンサイト | 高 | 低 | 高 | Depends | 高 |
| 2 | コロケーション | 中 | 高 | 中 | Depends | 高 |
| 3 | クラウド | 低 | 高 | 低 | Depends | 低 |

コロケーションに対してクラウドへのバックアップがコスト面で有利であることは

文献 [22]でも検証されている。クラウドを活用することで，コロケーションと比較して85%のコストを削減できることが示されている。ただし，クラウドに置かれたデータの保護や可用性はクラウド事業者の運用に依存するため，ユーザ自身がコントロールできないことが制約となる。この依存性を回避するために，汎用的な手法でクラウド間のデータ移行（マイグレーション）をする手法が提案されている [23]。

コストを下げることは性能の低下につながるため，そのバランスを適正に設計することが求められる。クラウドが普及する以前の従来型ディザスタリカバリサービスは，占有モデル（Dedicated）と共有モデル（Shared）のふたつのアプローチに分類される。占有モデルでは IT リソースが単一ユーザに割り当てられるため，コストと性能の両方が高い実現方式である。一方，共有モデルは複数のユーザにリソースを割り当てるため，コストと性能の両方が低下する。これに対し図 2.2 に示すように，クラウドコンピューティングによるディザスタリカバリが占有モデルの高性能と共有モデルの低コストの両方の利点を得るモデルであることを文献 [20]では示している。

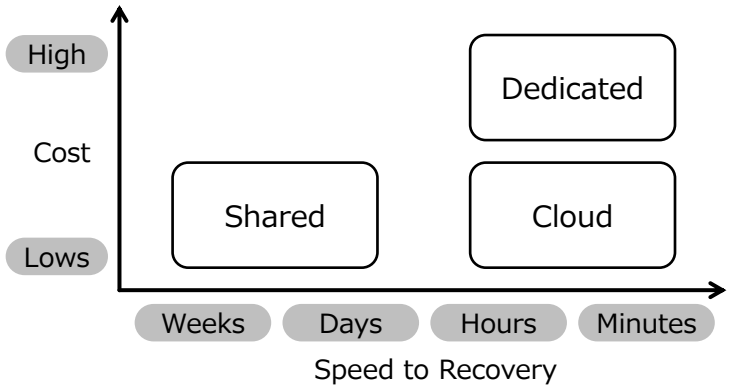



**図 2.2 ディザスタリカバリモデル**

## 2.3. データ保護システムの性能設計

データ保護のリスクを管理するために，既に二つのシステム性能指標が浸透している。図 2.3 にその考え方を示す。

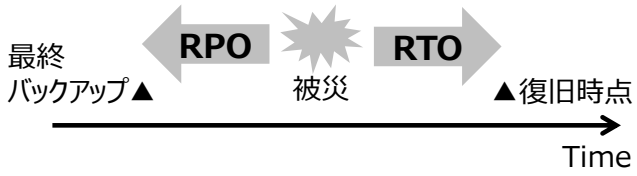



**図 2.3 性能指標（RPO と RTO）**

**Recovery Point Objective（RPO）:**

データがリストアされる最終バックアップ時点（最終的に失われるデータ量を反映する時点）までの時間の長さ

**Recovery Time Objective（RTO）:**

停止から操作の復旧までの時間の長さ

**表 2.4 ディザスタリカバリのレベル**

| Tier # | 項目 | RTO | RPO |
|---|---|---|---|
| 1 | Point in Time テープバックアップ | 2-7 日 | 2-24 時間 |
| 2 | リモートサイトへのテープバックアップ | 1-3 日 | 2-24 時間 |
| 3 | Point in Time ディスクコピー | 2-24 時間 | 2-24 時間 |
| 4 | リモートロギング（データベース） | 12-24 時間 | 5-30 分 |
| 5 | リアルタイムリモートロギング（RRDF） | 1-12 時間 | 5-10 分 |
| 6 | リモートコピー（ストレージコントローラ） | 1-4 時間 | 0-5 分 |
| 7 | リモートコピー＋フェイルオーバー | 0-60 分 | 0-5 分 |

データ保護システムは，これらの目標値を達成するように運用することが求められる。一例として文献 [24]では，表 2.4 に示すとおりデータの重要性に応じてディザスタリカバリのレベルを分けることが提案されている。クリティカルなデータについては分オーダの RPO および RTO が要求される一方，重要でないデータは 24 時間ごとのテープバックアップで十分であり，これらの使い分けを計画するべきである。なお，表中“Point

in Time”とは継続的にリモートサイトへデータを転送するのではなく，ある時点でまとめて処理するバックアップ手法を指す。

データ保護システムの運用においては性能を保証することと同様に，コストを下げることも重要な要件とされる。文献 [11]は，データ保護システムの設計を性能とコストの最適化問題と捉え，その自動計算ツールを考案した。図 2.4 に示すように，データ保護システムにおいて性能とコストはトレードオフの関係にあり，そのバランスを適正化することがひとつの課題となる。データ保護システムにおいて，想定を超える高い負荷にも耐えられる性能を備えた構成設計（オーバープロビジョニング）を行った場合，通常稼働時に過度のコストが発生するが，障害あるいは災害発生時に発生するコストを抑えることができる。逆に想定負荷を下回る性能にあわせた設計（アンダープロビジョニング）の場合は，通常稼働時には低コストで運用できるが，被災時にデータ消失やサービス停止といった追加コストが生じる可能性がある。データ保護システムの設計においては，こうしたトレードオフの関係を十分に理解し，最適なバランスを導出することが求められる。

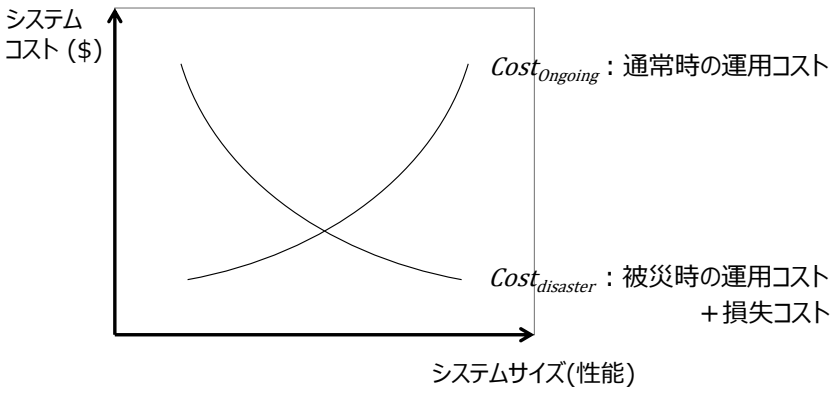



**図 2.4 性能とコストのトレードオフ**

文献 [11]では，コスト要件を入力として，適切なデータ保護ソリューションを出力するツールを考案した。RPO や RTO といったシステムパラメータを利用する代わりに，被災時に許容できる経済的損失（ロスコスト）を Penalty Rates パラメータとしている。

またバックアップサイトへのデータコピー方式として「ストレージコントローラ」「リモートミラーリング」「テープバックアップ」の３種類を想定し，各方式の振る舞いを簡易な数式によるモデルで定義した。さらに，同システムが想定するワークロード（データ書き込み負荷）を 11 種類の定数の組み合わせで表現した。図 2.5 に示すように，これらの入力を与えることで，要件を達成するバックアップ方式に加え，データ消失とデータ回復に要する損失を計算する最適化エンジンが提案された。

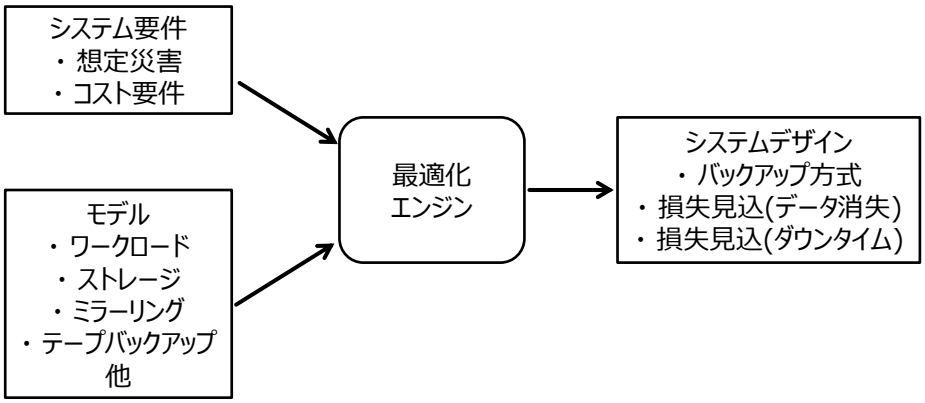



**図 2.5 ディザスタリカバリ設計ツールの構造**

これをより具体的な実装に近づけた研究もある [25] [26]。この研究は IBM 社とそのパートナー企業が提供する８種類のデータコピー技術から，要件に適合する方式を導出するツールを提案した。同ツールは想定する障害規模や RPO，RTO といった要件を，事前に構築したナレッジベースに照合することで，推奨するコピー手段やシステム構成を出力する。このツールによれば，IBM 社製ストレージやデータベース(DB2)，VMware 社のサーバ仮想化ソフトウェアに特化したシステム構成案を獲得することができる。

データコピー技術については，特に同期方式と非同期方式のふたつがある [27]。同期方式はより RPO と RTO のいずれについても高い性能目標を保証することができるが，複製処理にかかる時間がアプリケーションの性能を低下させるリスクがある。したがってデータ転送遅延を短時間に抑えるために必要なコストは高額になる。一方，非同期方式では複製処理がアプリケーション処理のオーバヘッドとはならないため，低いコスト

で運用することができるが，逆に保証できる RPO および RTO は劣化する。これらのトレードオフを鑑み，最適なコスト・性能バランスを実現することが重要である。

さらに，ディザスタリカバリではデータを生成するサイト（プライマリサイト）から距離を隔てた拠点（バックアップサイト）に複製したデータを保管するが，そのバックアップサイトの運営費が問題となり得る。文献 [22]では，バックアップサイトの運営手段を３段階に区別し，用途に応じて使い分けることでコストの抑制に取り組むべきであることを提案している。

**Hot Standby：**

バックアップサイトに常時稼働する待機系のサーバを設置し，継続的なミラーリングを実行する方式。復旧時にはこれを即座に稼働させて業務を継続する。待機系のシステム環境を一式用意しておく必要があるため，コストは高価となる。

**Warm Standby：**

データを同期あるいは非同期レプリケーションで継続的にバックアップサイトに複製する方式。復旧時にはアプリケーションサーバをはじめとするシステム環境を稼働するが，これらは通常時は稼働させずオフラインとしておく。そのため起動時にオンラインになるまで数分程度の時間が発生するが，その遅延を許容することでコストを抑制に寄与する。

**Cold Standby：**

データを継続的に複製するのではなく，スケジュールされたタイミングでまとめてバックアップを取得する。スケジュールに依存するが RPO は数時間から数日程度となる。また復旧時にはハードウェアを倉庫から持ち込む他，開発・テストなどの時間を要する場合があるため，RTO は数時間から数日程度となることが見込まれる。事業継続性や可用性の面では物足りない反面，安価なコストで実現できる方式である。

以上のバックアップサイト運営方針を表 2.4 に適用すると, #1 から#3 が Cold Standby, #4 と#5 が Warm Standby，#6 および#7 が Hot Standby に分類される [18] [21]。

## 2.4. データ保護システムの運用管理

文献 [17]は非同期リモートコピー方式を対象として，リカバリポイントを監視する技術を提案している。非同期方式では転送するデータがバッファに滞留し，遅延が発生する可能性がある。そこで，バッファの入口と出口でそれぞれ計測したライトシーケンス番号（複製するデータの断片に付与される順序番号）の差（タイムラグ）を調べることで，バッファ滞留時間を計測する。このタイムラグをリカバリポイントと見なすことで，監視を実現できることを示している。

## 2.5. 実証基盤

クラウドを活用したディザスタリカバリ研究に利用されてきたプラットフォームを以下にまとめる。

(1) RUBiS

RUBiS [22]は，ディザスタリカバリとサービスレベルに基づいたコスト最適化の検証を目的としたアーキテクチャであり，バックアップサイトにクラウドを採用している。通常運用時はレプリケーションモードで動作し, 継続的にデータをクラウドへ転送する。被災時にはフェイルオーバーモード（復旧モード）となり，実行系をクラウドに切り替えて業務継続を試みる。フェイルオーバーモードでアクティブとなるクラウド側のリソースはレプリケーションモードでは使用されない。そこで通常稼働時には余剰資源を他の用途に貸し出すといったコスト削減の仕組みを提案している。

(2) PipeCloud

PipeCloud [28]は，データ複製に独自技術である“Pipeline Replication”を活用したプラットフォームである。非同期コピーの性能と同期コピーの一貫性（Consistency）の両立を図るため，WEB サーバの応答処理，データベースの更新処理，クラウドへの転送処

理を並行実行する。それぞれの処理のタイミングを調整し，最終的な WEB クライアントへの応答を待たせることで全体の性能を上げる仕組みとなっている。

(3) Remus

Remux [29]は Xen ハイパーバイザベースのクラウドプラットフォームでありながら，データレプリケーションにはストレージを採用している。データ転送の過程で特定のタイミングにおいて“チェックポイント”を設け，それまでのデータをすべてバックアップ側の RAM に格納する。すべての書き込みデータを転送するため，より高品質なデータ転送ネットワークが必要となる。

(4) Romulus

Romulus [30]は，KVM ハイパーバイザをベースとしたディザスタリカバリシステムであり，Remus の改良版に位置づけられる。Romulus はデータ複製用のバッファを新たに追加することで，Remus の信頼性における問題（Single Point of Failure）を解決している。またチェックポイントごとにすべてのデータを転送するまで，アプリケーション性能が低下する問題を解決するため，ネットワークの出口にもバッファが追加された。

Romulus の 7 ステップによる耐障害アルゴリズムは，データだけでなく VM の整合性を保証する基本的なプロセスを示しており，広く引用されている。

(5) Kemari

Kemari [31]は，仮想マシンの同期によるクラスタ技術である。Xen ハイパーバイザをベースとしたプラットフォームであり，イベントを契機に同期を行うことで，アプリケーションに依存せずサービスを継続できる。

上記以外にも，Taiji [32]，SecondSite [33]，HS-DRT System [34]，Disaster-CDM [35]，などのプラットフォームが開発されてきた。

## 2.6. 本研究の位置づけ

本研究は，性能要件を達成するデータ保護システムの運用管理に関し，特に RTO ではなく RPO を達成する技術を提案する。RPO は既に普及している性能指標であるにも関わらず，その測定を実現する汎用的な技術が確立されていない。文献 [17]は，ストレージの内部パラメータであるシーケンス番号に着目したリカバリポイント監視技術を提案している。しかしながら，このようなパラメータは外部には非公開であるため，ストレージを供給するベンダしか利用できないという制約がある。加えてコピー方式はベンダごと，あるいは製品ごとに独自の設計であるため，それぞれに特化した測定機能の実装が都度必要となる。したがって，リカバリポイントの監視にあたっては，製品の内部パラメータを活用しない汎用的な手法の確立が課題となる。本研究では，書き込みデータ量などの標準的な監視メトリックを使用することで，ベンダの製品仕様に依存しないリカバリポイント監視技術を提案する。

また 2.3 節に述べたとおり，データ保護システムの計画にあたっては，性能とコストの最適化問題に取り組まなければならない。文献 [11]はこの問題に対し，システムの振る舞いをモデルで表現し，計算によって最適解を導出するアプローチを提案した。同研究は経済指標（データ消失などの損失コスト）を入力として採用すべきシステム構成を判定するが，この損失コストはシステム設計段階ではなく，より上流の経営計画の段階で考察するべきパラメータであることに注意が必要である。すなわち同技術は，システム設計より上流工程である経営計画段階で活用する技術に位置づけられる。システム設計段階では RPO あるいは RTO といった具体的な指標によって構成を設計しなければならないため，本研究では，RPO を達成するシステム構成の設計に取り組む方針とした。本方式では，RPO を達成しながらも，最小のコストでデータ保護を実現するシステム構成を導出することを可能とする。

文献 [25] [26]は IBM 社や VMware 社の製品を活用したシステム構成を計画するツールを開発したが，その実行にあたっては，各々の製品仕様に基づくナレッジを事前に構築しておく必要がある。しかしながら，製品の挙動や制約を構造化したナレッジを策定

できるのは，その内部仕様を熟知した開発関係者に限られるため，これは一般に応用できる手法であるとは言いがたい。加えて，この手法では製品の更新にあわせてナレッジを更新しなければならないことも問題となる。これに対し，本研究は製品仕様に依存しない汎用的なシステム設計方式を提案する。前述のとおり，本研究では製品やサービスの仕様に特化せず，標準的なシステム管理パラメータだけを利用することで，様々な実装に応用できる手法の確立を目指す。

# 3章　情報システムの変遷

## 3.1. マクロトレンド

本研究は，様々なデータを取り扱う計算機システムを対象とする。以下，これを情報システムと記す。情報システムを具現化する形態は，技術の進化，取り扱うデータの種類と量，アプリケーションや利用目的などの要因によって時間とともに変化してきた。本章ではこれらの変遷をマクロな視点で整理した上で，本研究が対象とする領域を明確にする。

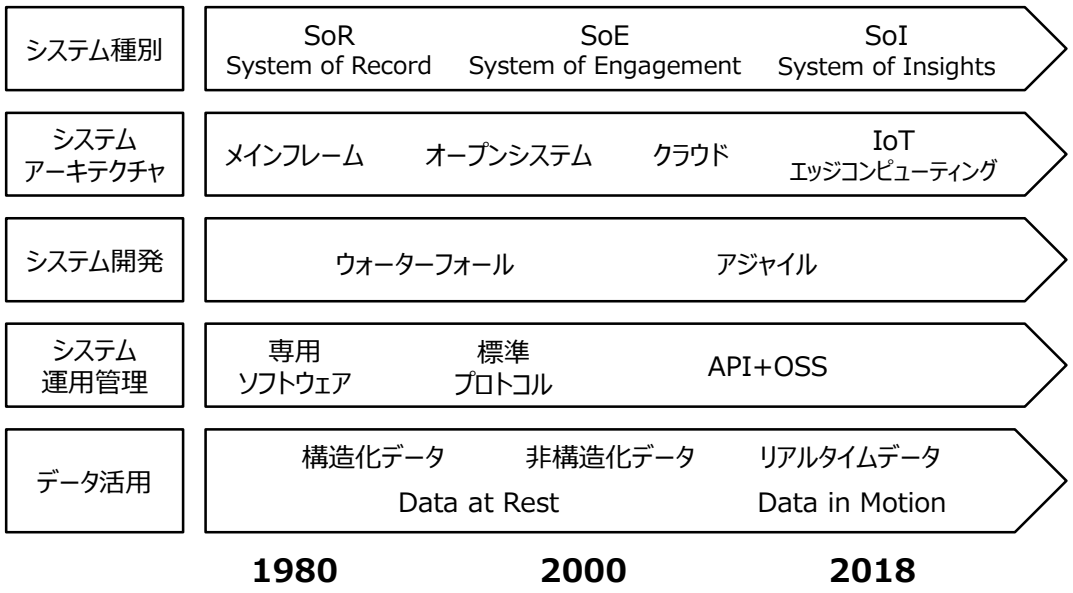



**図 3.1 情報システムのマクロトレンド**

図 3.1 に情報システムの変遷を示す。情報システムは，利用目的や技術の変化に沿って，システムアーキテクチャ，開発と運用手法，取り扱うデータなどそれぞれが進化してきた。情報システムは企業や公共団体などの様々な基幹業務に利用されてきたが，従来の用途は，取引情報や顧客情報などの記録と参照であった。こうした目的で開発された情報システムは“System of Record（SoR）”と呼ばれる [36]。これに対し，企業と顧客，あるいは個人と個人など，様々な関係をつなぐことを目的としたシステムは“System

of Engagement（SoE）”に分類される [36]。一例として，WEB サービスや SNS の分野では利用者の趣向や位置情報に応じた広告のカスタマイズにより，その効果であるエンゲージメント率を高めようとする。このように企業と利用者の関係を結びつけることを狙いとした情報システムを SoE と総称する。

**表 3.1 System of Record と System of Engagement**

| # | 項目 | SoR (System of Record) | SoE (System of Engagement) |
|---|---|---|---|
| 1 | 目的 | データを確実に記録 | 人やモノをつなげて<br>新たな価値を創出 |
| 2 | アプリ | 勘定系，ERP 他 | SNS，WEB サービス他 |
| 3 | 要件定義 | 要件が固定的であるため<br>事前要件定義が可能 | 市場や顧客の状況に応じて<br>常に要件が変化 |
| 4 | データ構造 | 構造化データ，RDB | 非構造データ |
| 5 | システム開発 | ウォーターフォール型 | アジャイル開発 |
| 6 | 用途・期待 | 信頼性・可用性，<br>トランザクション性能，<br>データの完全性，<br>従来システムからの移行，<br>従来システムとの互換性 | システム更改の容易性・俊敏性<br>（アジリティ），<br>API によるシステム間接続，<br>短期開発や継続的改善 |

SoR と SoE では，求められる要件が異なる。SoR の運用においては，記録を失わないことに加え，いつでも安定して記録を参照できること，すなわち信頼性や可用性を最も重要と位置づける。金融機関が預貯金の記録を消失したり，預金残高を照会できないといったことがあってはならないため，これは当然の要件といえる。一方で SoE では利用者の要求にいち早く応えるためのアップデートや，経験価値を高めるためのインタフェース改善，あるいはアプリケーションの不具合修正など，開発成果を次々にリリースすることが求められる。SoR が信頼性を担保するために入念な事前設計やテストを行い，数ヶ月に一度のシステム更改といった運用を前提とするのに対し，SoE は計算リソース

やストレージリソースを需要に応じて増減させながら，毎日のようにソフトウェアを更改する。このように，SoR と SoE ではシステムアーキテクチャおよび開発，運用手法に大きな違いがある。本章ではこれらの違いについて整理する。

2010 年にはビッグデータの概念が登場し，膨大な量のデータに基づいた分析を通じて新たな知見を獲得しようとする取り組みが活発に行われるようになった。これを実現する情報システムは“System of Insight（SoI）”に分類される [37]。SoI には，多種多様なデータを集め，蓄え，整理して分析する機能に加え，データそのものや分析結果を安全に公開するインタフェースの実装などが求められる。分析するデータは，SoR に記録されたビジネスデータならびに WEB などの公開情報だけでなく，フィールドに設置されたセンサおよびカメラから集まるデータも対象となる。様々なモノ（Thing）をネットワークに接続し，センサやアクチュエータによって監視・制御する概念は“Internet of Things（IoT）”と呼ばれる。企業では，これまでネットワークに接続されていなかった設備や機器，人の振る舞いをセンサあるいはカメラによってデジタルデータに換え，監視・制御可能となるように自社の仕組みを変革する「デジタルトランスフォーメーション」に取り組もうとしている。これまで把握できなかった事象をデジタルデータで可視化し，様々な洞察を得ることでコスト低減や売上拡大といった経営課題を解決できるようになることが期待されている。

米国の調査会社 IDC は，こうした SoE や SoI を実現する新しいシステムを「第３のプラットフォーム」と定義している [38]。その主要構成要素はモバイル，ソーシャル，ビッグデータ，クラウドの４つであり，それらが登場する以前の SoR を構成するプラットフォーム（第１，第２のプラットフォーム）とは別の市場を形成すると予測している。

本章では以下，システムアーキテクチャ（3.2 節），システム開発（3.3 節），運用管理（3.4 節），データ活用（3.5 節）の変遷を順に整理し，さらにこれからの社会課題解決に向けた情報システムの役割を考察する（3.6 節）。これらの整理に基づき，本研究のスコープを明確化する（3.7 節）。

## 3.2. システムアーキテクチャ

### 3.2.1. メインフレーム

メインフレームは 1980 年代に普及した汎用大型計算機である。その歴史は長く，多様な進化を遂げてきたが，基本的にはメーカー独自規格のハードウェアならびにソフトウェアによって性能や信頼性を高めてきた計算機システムである [39]。性能と信頼性に優れる一方，その構築や運用保守はメインフレームベンダに依存する以外の選択肢が限られ，販売価格や保守費用が非常に高価であると言われている。その一方で複数ワークロードの並列処理や大規模バッチ処理での性能と信頼性の高さから，特に企業の基幹業務に採用されてきた。

計算機システムに対する要求性能がさらに高まると，ホストコンピュータの数を増やして対応した。さらにワークロードの増加にあわせてデータ量が増えてくると，今度はストレージ利用効率が課題となった。こうした課題の解決に向け，計算処理を実行するホストコンピュータと，データの記録と保管を行うストレージを分離する SAN（Storage Area Network）のアーキテクチャが登場した。SAN によりストレージサブシステムは複数のホストで共有され，容量利用効率の向上とデータ集約による運用管理負担の軽減が図られるようになった。ストレージサブシステムは，数千台規模のハードディスクドライブ（HDD），読み書き要求を処理する数十台～数百台のプロセッサ，ディスクキャッシュ，サーバとのインタフェース部，ディスクとのインタフェース部とこれらを結合する機構などの要素で構成する。SAN で最も多く普及している接続形態は FibreChannel（FC）である。FibreChannel は ANSI T11 委員会で標準化されたネットワーク技術であり，SCSI プロトコルや FICON（Fiber Connection）プロトコルを用いたデータの読み書きが可能である。FICON はメインフレーム用の入手出力インタフェースである ESCON（Enterprise System Connection）互換のデータ入出力プロトコルである [40]。

### 3.2.2. オープンシステム

メインフレームがメーカー独自規格のコンピュータをベースに開発されたシステム

であるのに対し，公開された仕様に準拠したハードウェアおよびソフトウェアを利用し，またその要件を満たすために様々なベンダの製品を組み合わせて構築するシステムをオープンシステムと称する。データセンタに構築された基幹系業務システムは，特にUNIX サーバとデータベースで構成するケースが多い。

90 年代にはイーサネットによる LAN（Local Area Network）技術が普及し，PC（Personal Computer）が相互に接続されるようになった。PC は業務システム（サーバ）にアクセスするための端末（クライアント）としての役割を担った。このサーバとクライアントで構成するシステムが，クライアント・サーバシステムである。同システムにより，データセンタでの集中処理と，クライアントの分散処理を組み合わせて効率的な運用が可能となった。

1990 年代後半になると，計算処理を実行するサーバと，データの記録と保管を行うストレージを分離してネットワーク接続する実装形態がさらに発展した。表 3.2 に示すとおり，サーバにストレージを直接接続する DAS（Direct Attached Storage），ストレージ専用のネットワークで接続する SAN（Storage Area Network），LAN 経由でサーバにファイルストレージを接続する NAS（Network Attached Storage）の中から，システムの目的やデータ量に応じて適切な実装を選択できるようになった [40]。

**表 3.2 ストレージシステムの実装**

| # | 種別 | 実装形態 | メリット | デメリット |
|---|---|---|---|---|
| 1 | DAS | サーバとストレージを SCSI や FibreChannel で直結 | 導入コストが低い | 容量利用効率が悪い<br>運用管理が面倒 |
| 2 | SAN | 複数のサーバと複数のブロックストレージをネットワークで接続 | アクセス性能が高い<br>容量利用効率が良い<br>運用管理が容易 | 導入コストが高い |
| 3 | NAS | ファイルストレージを LAN に接続，複数のクライアントで共有 | 導入が比較的容易<br>容量利用効率が良い | アクセス性能が LAN に依存，ネットワーク性能低下の要因 |

SAN を構成するネットワークの実装にはさらにいくつかのバリエーションがある。前述のとおり，SAN の普及段階では Fibre Channel（FC）がネットワークに利用された。その後，LAN とは別にストレージ専用のネットワークを実装するコストや，その運用管理に FC の専任技術者を設ける必要があるといった課題を解決するため，FC ではなく Ethernet 上にストレージネットワークを構築する技術が開発された。iSCSI （Internet Small Computer System Interface）と FCoE（Fibre Channel over Ethernet）である。iSCSI は TCP/IP 上で SCSI による I/O プロセスを実行するプロトコルスタックである。これに対して FCoE は，DCB（Data Center Bridging）により Ethernet 上のロスレス通信やマルチパスなどの機能を補完することで，FC フレームの送受信を Ethernet 上で実現する規格である [41]。

次に，システムアーキテクチャの進化に大きな影響を与えた技術のひとつがサーバ仮想化である。それまでの計算機のインスタンスは，物理サーバ 1 台にひとつの OS を稼働する形態であったが，仮想化により複数のインスタンスを 1 台の物理サーバで稼働できるようになった。仮想化技術は①パーティショニング方式，②仮想 OS 方式，③ハイパーバイザ方式の順に開発されてきた。①パーティショニング方式の実装である LPAR（Logical Partitioning）はメインフレームでも実装されていた機能であり，1 台のサーバを論理的に複数の区画に分割する。この論理区画をあたかも 1 台の物理サーバとして利用できることが LPAR 方式の特徴である [42]。これに対して②仮想 OS 方式はサーバにホスト OS を稼働し，その上に仮想サーバのインスタンスである「ゲスト OS」を起動する。複数のゲスト OS を起動することで，1 台の物理サーバに複数の仮想サーバを稼働させることができる。③ハイパーバイザ方式は，ホスト OS ではなく専用の仮想化機構を提供するソフトウェアである [43]。図 3.2 にはこれらの実装形態の違いと，ストレージリソースの仮想化を例示した。一般にパーティショニング方式は物理リソースの一区画（論理デバイス）を LPAR に割り当てていく機構である。これに対して仮想 OS 方式とハイパーバイザ方式では，それぞれの仮想化機構が仮想サーバ（VM：Virtual Machine）に割り当てる仮想ディスクをイメージファイルで生成する。これらの方式で

は物理資源をすべて隠蔽し，仮想化された単位で運用する点が特徴である。

こうした仮想化技術の進歩により，物理ハードウェアとサーバインスタンスを分離できるようになり，柔軟かつ機動的なシステム運用が可能となった。

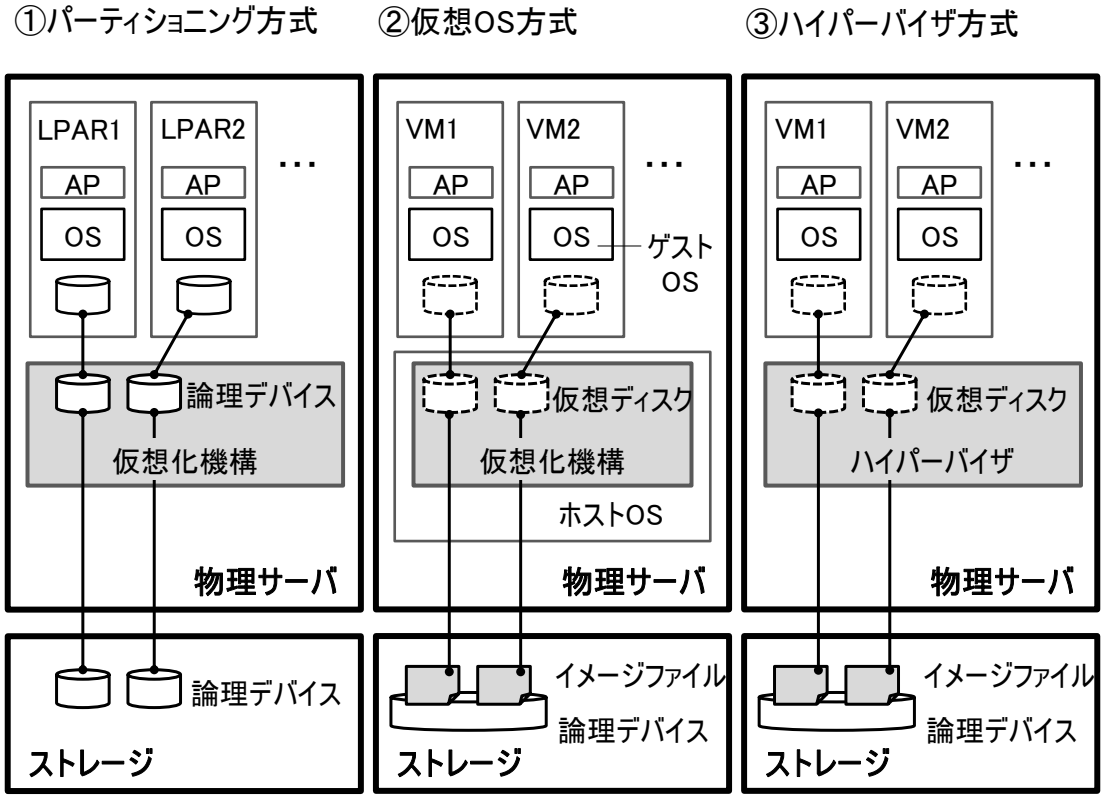



**図 3.2 サーバ仮想化方式**

### 3.2.3. クラウドコンピューティング

仮想化とともに，サーバやストレージを専用ハードではなくソフトウェアで実装する Software-Defined 技術が進歩すると，クラウドサービスが実用段階に至った。クラウドサービス事業者は多数の汎用サーバを常時稼働する大規模なデータセンタを設営し，またそれらを広帯域ネットワークで接続し，ユーザの要求に応じて必要な数，必要な量のシステムリソースを供給できるようにした。その用途は様々であるが，例えば多数の WEB サーバをクラウドに立ち上げ，これらを並列稼働させることでインターネットからの大量のアクセスを受け付けることが可能となる。こうした WEB システムは，SNS

やオンラインゲームなどの膨大な数のリクエストを処理するコンシューマ向けサービスの用途に適している。

クラウド事業者のデータセンタから計算機資源を調達する実装形態は，“オフプレミス”に分類される。この“オフプレミス”に対して，企業が所有する設備に自社利用目的に限定したクラウドを自ら構築・運用する実装を“オンプレミス”と呼ぶ。自社の運用コストが発生するが, セキュリティルールにより社外に持ち出せないデータの保管に適するほか, 長期的にはクラウド事業者に支払うコストよりもコスト面で有利になる可能性があるといったメリットを得られる可能性がある。さらにこれらの方式を適材適所で組み合わせ，双方のメリットを得ようとする方式がハイブリッドクラウドである[44]。

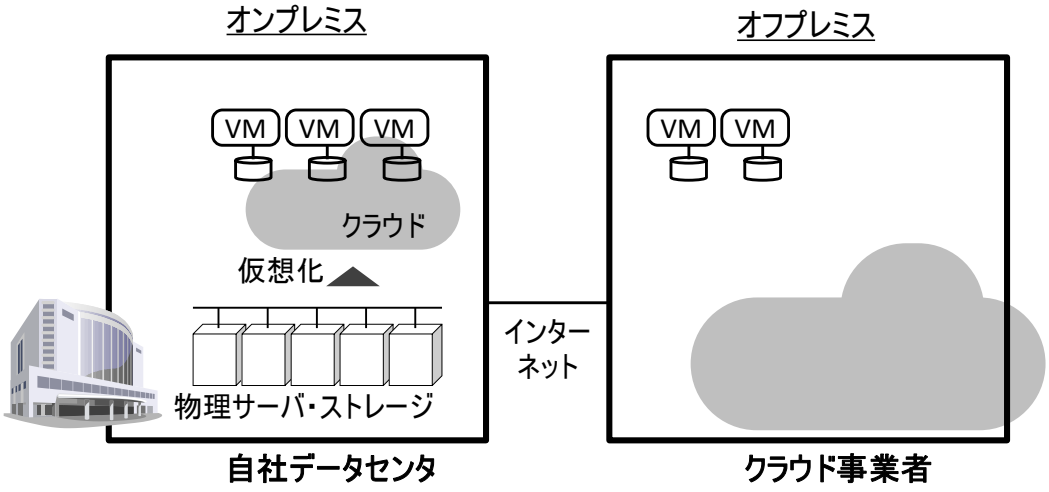



**図 3.3 オンプレミスとオフプレミス**

さらにクラウドサービスは，その提供内容により３段階に分類できる。IaaS（Infrastructure as a Service）は，仮想サーバのインスタンスやストレージをはじめとする計算機資源を提供するサービスである。サーバの性能やストレージ容量などのスペックをユーザが指定し，オンデマンドサービスとして利用できる点が特徴である。これに対して PaaS（Platform as a Service）は OS などの基本ソフトだけでなく，アプリケーションの開発環境や実行環境などのソフトウェア一式をサービスとして提供する。さらに

SaaS（Software as a Service）はアプリケーションソフトウェアをクラウドで提供するサービスである。メールや文書作成ソフトなどの SaaS は広く普及している。

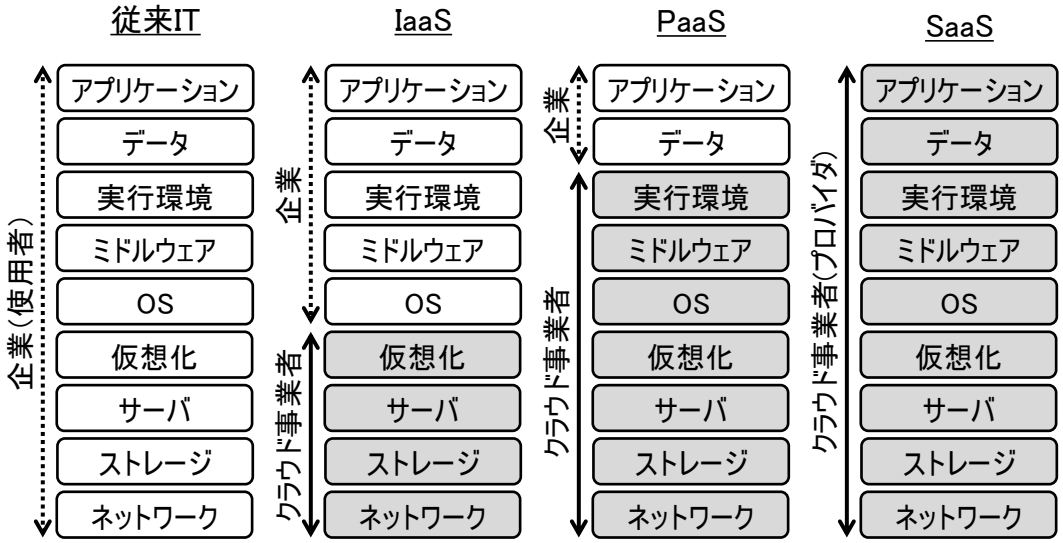



**図 3.4 クラウドサービスの提供範囲**

### 3.2.4. エッジコンピューティング

企業情報システムがクラウドでの集中処理に移行する一方で，フィールドに存在するデバイスのより近くに設置された計算機に処理を分散するエッジコンピューティング技術の開発が進んでいる。その背景には IoT（Internet of Things）の発展がある。実世界のあらゆるフィールドに設置されたセンサやカメラといったデバイスがネットワークに接続されるようになり，それらが生成したデータの分析により潜在的なリスクを発見する，あるいは状況に応じてそれらのデバイスを制御するといった IoT ソリューションに期待が高まっている。その実現にあたり，特にリアルタイム処理を必要とする場面では，クラウドへのデータ転送遅延が問題となる。例えば音声対話アプリケーションでは，秒単位の遅延が経験価値を大きく損なう。さらに自動運転や遠隔医療など，リアルタイムで画像処理し，応答しなければならないアプリケーションでは，ミリ秒単位の遅延を許容できない可能性もある。

そこでエッジコンピューティングでは，データ処理のポイントをクラウド，エッジ，

エンドポイントの３段階に分けて処理を分散する [2]。図 3.5 にクラウド，エッジ，エンドポイントの位置づけを示す。

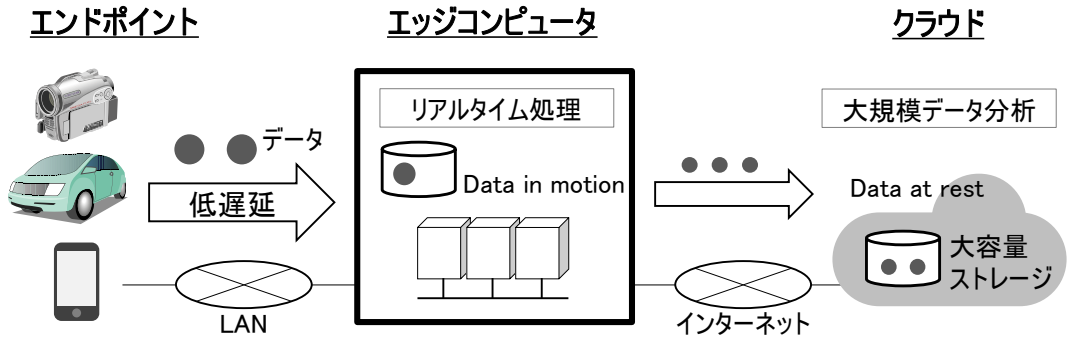



**図 3.5 エッジコンピューティング**

**クラウド**

クラウドにはデータを蓄積し，機械学習や深層学習といった技術によりデータ分析するための基盤とする。

**エッジコンピュータ**

企業のサーバルームやフィールドに設置されたサーバなど，地域での応答時間を短縮するために設置された小規模データセンタを含む。転送遅延を許容できないアプリケーションでは，データをクラウドに送る前に処理を済ませなければならないことがある。そのためエンドポイントに近い場所にエッジコンピュータを配置し，エンドポイントから集めたデータを処理する機能を持たせる。クラウドへの送信データを圧縮もしくは削除するほか，データの特徴量を抽出して送信データ量を削減するといった処理をリアルタイムで行い，データ転送を効率化する。あるいは自動翻訳の中間処理などアプリケーションをエッジで実行し，応答時間を短縮する試みもある。

**エンドポイント**

エンドポイントは，センサやカメラあるいはそれらを搭載したモバイル機器，あるい

は産業機器や建設機械など，ネットワークの末端に接続されたすべてのデバイスが相当する。今後はカメラや家電など，ネットワークにつながりデータを生成する IoT 機器の数が爆発的に増えることが予想されている。

エッジコンピューティングの採用にあたっては，どの処理をどのポイントで実行するかを決定する，全体のシステム設計が課題となる。IT 技術の進歩は早く，特に計算機やネットワークの高性能化は顕著であるため，クラウド・エッジ・エンドポイントの性能バランスは常に変化する。したがってエッジコンピューティングでは，コンポーネントの処理性能とデータ量，計算量を勘案しながら，要件を満たすようにシステム全体を設計することが求められる。

### 3.3. システム開発

前節で述べたように，情報システムの利用目的の変化や技術の進歩によってアーキテクチャが変化し，その度にシステム開発や運用管理手法も様々な変遷をたどってきた。

#### 3.3.1. ウォーターフォール型

第一のプラットフォームにおけるシステム開発は，メインフレームを供給するベンダに依存していた。これに対して，第二のプラットフォームではサーバやストレージ，OS をはじめとするソフトウェア群を適材適所に組み合わせてシステムを構築する“システムインテグレーション（SI）”を前提とした開発手法が主流となった。システムインテグレーションには装置の仕様やネットワーク設計などの専門スキルを有するため，そのプロフェッショナルであるシステムエンジニアを抱える SI サービス事業者に開発を委託することが多い。その開発プロセスは要件定義から設計開発，テストを段階的に進めるウォーターフォール型である。ウォーターフォールによる開発期間は一般的に数ヶ月から 1 年程度を要する。特に信頼性や可用性，ピーク性能，セキュリティなどの要件を達成するために，ネットワーク構成設計やサーバ・ストレージ機器の選定，そのテストや性能評価に十分な時間をかける。テスト工程ではモジュール単位の単体テストから結合テスト，システムテストと段階を追って入念な検査を行うことで，信頼性の高いシス

テムを構築する。

### 3.3.2. アジャイル型

事前にシステム要件を定義し，それを達成するように作り込むウォーターフォールに対して，SNS や WEB あるいはモバイルアプリケーションのように，頻繁にアップデートを繰り返しながら完成度を高めていく開発技法をアジャイル型と呼ぶ。アジャイル型開発では，少数のチームが２週間程度の短い開発期間（スプリント）を単位として小規模の成果をリリースし，そのサイクルを繰り返しながら段階的に改良していくアプローチである [45]。

SoR から SoE への遷移に伴って，従来のウォーターフォール型開発の問題点が浮き彫りとなり，リリースサイクルの短いアジャイル開発へのシフトが進んでいる。ウォーターフォール開発は仕様変更や前工程へのフィードバックを想定していないため，新技術の取り込みや事業環境の変化，ユーザ要求への対応など次々に発生するニーズへの対策が困難である。アジャイル開発はこうした問題を解決し，変化の早いマーケットに即応できる技法として採用する企業が増えている。

**表 3.3 ソフトウェア開発技法の比較**

| # | ウォーターフォール | アジャイル |
|---|---|---|
| 1 | ● 時間やコストよりも品質重視<br>● 柔軟性がなく，仕様変更困難<br>● 膨大なドキュメントが必要 | ● 品質が不安定<br>● 実験を通じて仕様をカスタマイズ<br>● アジャイル開発の経験が必要 |

いずれの開発手法においても，要件に適合するシステムサイズを決定する「キャパシティプラニング」は特に重要であるが，その実現に向けたアプローチがやや異なる。ウォーターフォールで開発するシステムは，基幹業務など信頼性や性能などの品質を重要視する上，その更改頻度が多くないため，設計段階でピーク時点の負荷を処理できるだけの十分な量のシステムリソースを設ける傾向がある。対してアジャイル型開発では，

負荷の増減にあわせて CPU やストレージなどのリソース量を調整しながら，変化に対応していく運用管理の確立が課題となる。

### 3.4. システム運用管理

情報システムを安定稼働させるためには，運用管理の役割が重要である。運用管理プロセスの PDCA（Plan-Do-Check-Action）サイクルを適切に回すことで，事前に設計されたとおりの振る舞いであるかどうか判断し，想定外の事象に対策することができる。

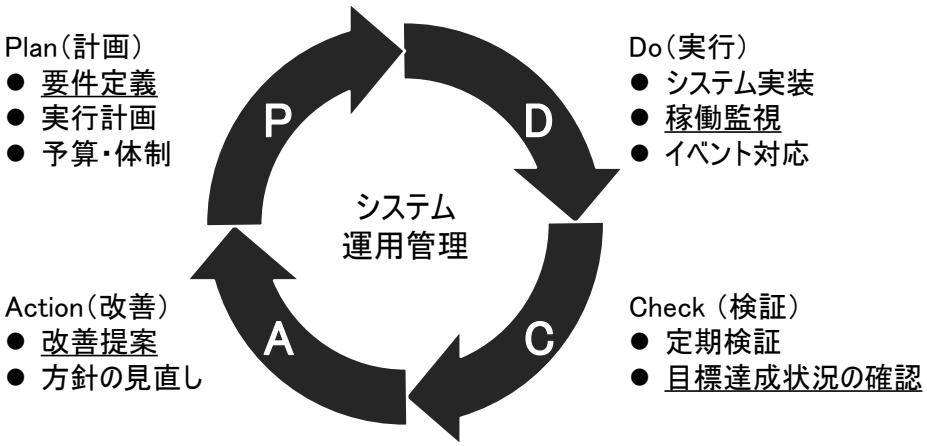



**図 3.6 システム運用管理の PDCA**

Plan（計画）フェーズでは，運用する情報システムの要件を定義する。市場ニーズの変化，ユーザ数やアクセス数など負荷の増減，新技術の導入など様々な要因で発生する経営環境の変化に応じて，システムキャパシティや性能，セキュリティなどの要件を新たに定義する，あるいは見直すフェーズである。Do（実行）フェーズでは，実装したシステムの日常的な稼働監視を行う。ハードウェア障害や通信エラー，過負荷によるサービスレベルの低下，セキュリティトラブルなどの問題をできるだけ早く検知し，対策を行う。そのために様々なツールを用いた自動検知のしかけを開発することが望ましい。Check（検証）フェーズは，稼働しているシステムが当初の要件を満たすように動作しているか，様々な観点から検証するフェーズである。当初想定したユーザアクセス数やデータ量と比べて，実際の稼働実績が大きく外れていないか，あるいは当初想定してい

なかったセキュリティの脅威への対策が必要かどうかといった考察を通じ，稼働中のシステムの妥当性を検証する。Action（改善）フェーズでは，検証結果に基づいてシステムリソースの増設や新機種へのリプレース，新機能の導入といった改良を提案する。

図 3.6 の PDCA を実行するためには，運用管理の仕組みやプロセスをシステムとして実現することが必要である。本稿ではこれを情報システムと区別するために，“運用管理サブシステム”と称する。運用管理サブシステムは，情報システムを構成するコンポーネントのひとつに位置づけられる。

運用管理サブシステムの構成要素には (1)業務プロセス，(2)管理ソフトウェア，(3)管理データ，(4)チームがある。事象を検知するための管理ソフトによる自動監視機能や，そのデータの保管と活用，さらに事象に対する管理部門のチームを編成し，それら業務プロセスを定義してそれらをシステム化することで，サービス品質の向上に寄与する。以下の節ではこれらの要素について個別に解説する。

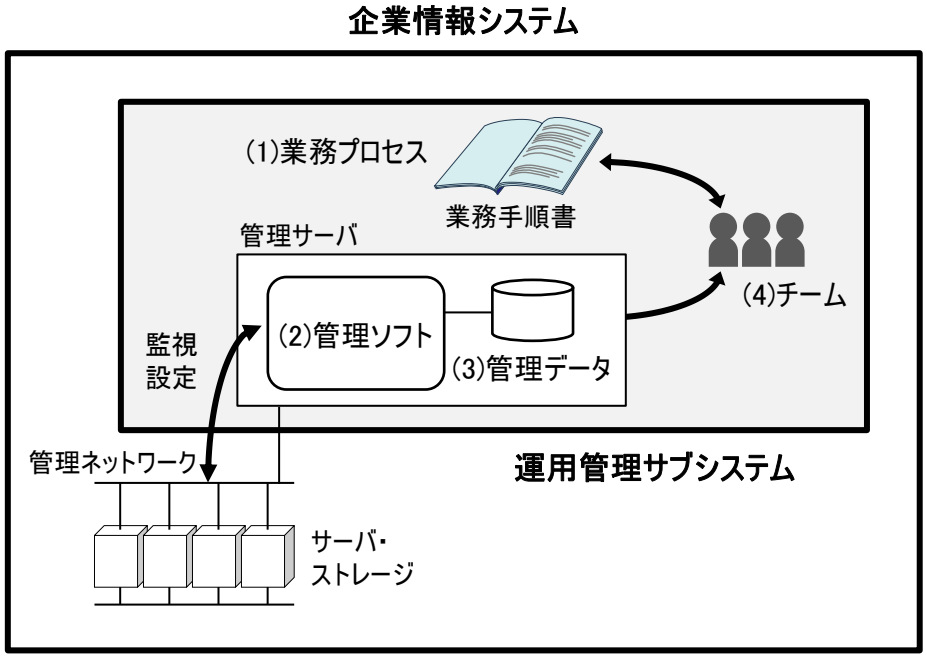



**図 3.7 システム運用管理サブシステム**

### 3.4.1. 運用管理業務プロセス

第一のプラットフォームでは運用管理もメインフレームベンダが用意したソフトウェアに依存し，それ以外の選択肢は限られていた。これに対して第二のプラットフォームが普及すると，マルチベンダ機器の組み合わせや構成の複雑化に関する問題が顕在化し，その運用管理の考え方が盛んに議論された。1980 年代には英国政府の活動を発端として，情報システムにおける運用管理のベストプラクティスやフレームワークを書籍にまとめた ITIL（Information Technology Infrastructure Library）が発刊された。その後，特に ITIL の考え方のコアとなる「サービスサポート」と「サービスデリバリ」を中心に改訂された ITIL v2 が 2001 年に発刊された [46] [47]。

サービスサポートは「インシデント管理」「問題管理」「構成管理」「変更管理」「リリース管理」の 5 つのプロセスと「サービスデスク」の 1 つの機能による構成であり，日々の IT 運用手法とその着眼点がまとめられている。サービスデリバリでは，「サービスレベル管理」「IT サービス財務管理」「可用性管理」「IT サービス継続性管理」「キャパシティ管理」の 5 つのプロセスについて，主に IT サービスの計画と改善手法について説明されている。

本研究は，ITIL におけるサービスデリバリの業務プロセスを実現するための技術に位置づけられる。さらに，システムの能力（キャパシティ）を適正なサイズに保つための運用技術に関する。

2000 年代には管理者を介さずに運用管理 PDCA を遂行する自律化技術の開発が盛んに行われた。製品の振る舞いやベストプラクティスに基づいて If-then ルールを整備し，これをナレッジベースに統合してシステムパラメータの自動調整や障害からの自動回復が図られた。

### 3.4.2. 運用管理ソフトウェア

前節の運用管理業務プロセスを実行するツールのひとつが運用管理ソフトウェアである。特にサーバやストレージ，ネットワークの振る舞いを監視し，異常を検知するためには，定常的にそれらの稼働状況をモニタリングするツールが必要になる。第一のプ

ラットフォームでは，メインフレームベンダがやはり独自の運用管理ソフトを供給していた。これに対し第二のプラットフォームであるオープンシステムでは，同システムを構成する機器の供給元ベンダが複数にわたるため，統一的なアーキテクチャが存在せず，システム監視は複雑なものとなった。そこでまずいくつかのソフトウェアベンダによって，ネットワーク監視専用のソフトウェアが開発された。情報システムの構成機器はネットワークに接続されているため，監視ソフトから Ping や Telnet，SSH などの手段を併用した状態監視が行われた。特に機器の状態や負荷，イベント情報を送受信するためのプロトコルには SNMP（Simple Network Management Protocol）が国際標準として採用されている。SNMP をサポートする管理ソフトは，情報システムを構成する機器から稼働情報やイベントを同プロトコルで受信する。管理ソフト側では数々の機器から受信した稼働データをリポジトリに格納し，GUI への出力や閾値検査などの機能を提供する。

クラウドが普及し，システム運用管理の対象が物理機器からサービス単位になると，管理ソフトに求められる機能も変化した。クラウドからは稼働情報がサービスとして提供されるため，個々の機器から SNMP などのプロトコルでデータを取得する機能を設ける必要がなくなる。一例として Amazon Web Services（AWS）では，利用中のサーバやストレージの稼働状況を Cloudwatch サービスで提供している [48]。そのインタフェースは Cloudwatch 専用の API として公開されているため，データを取得する管理ソフトは同 API で必要なデータを採取すれば良い。同サービスは CPU 利用率やメモリ使用率などの基本的なメトリックだけでなく，利用者自身が簡易な設定により必要なメトリックを出力対象に追加することができるカスタマイズ機能も備えている。

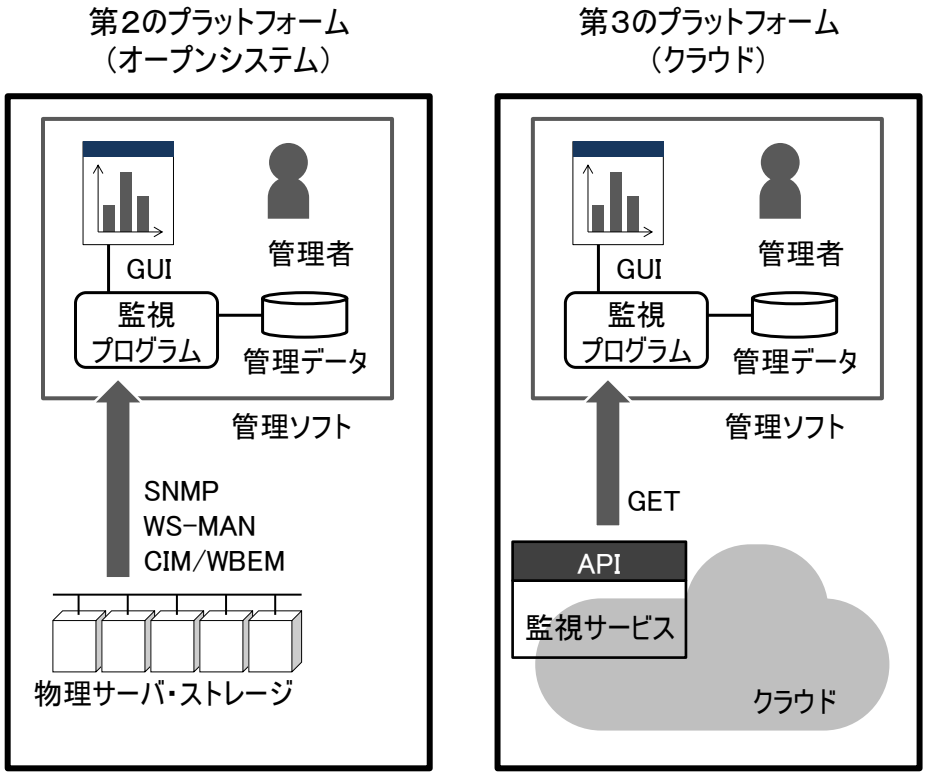



**図 3.8 システム管理ソフトウェアの実装**

運用管理ソフトウェアがシステムの挙動を適切に監視することで，閾値超過や障害などのイベントを検出し，即時の対策のきっかけをつくりだすことができるようになる[49]。

### 3.4.3. 運用管理データ

前節で述べたとおり，情報システムの稼働実績は，定義された監視メトリックごとの時系列数値データで出力される。出力するデータの時間間隔が短いほど高精度かつ即時性の高いモニタリングが可能になるが，通信負荷およびデータ量が増えることが課題となる。逆に間隔が大きいほど通信負荷や保管するデータ量が小さくなるが，高精度で低遅延の運用管理は行えない。そのため運用する監視データのサンプリング間隔を，要件を達成するように適切に制御することが求められる。

前節に挙げた Cloudwatch の例では，最短 1 秒から最長 1 日まで取得するデータのサ

ンプリング間隔を調整することができる [48]。データのサンプリング間隔が短いほど精度は高いが，データ保存期間も短縮されるため，高頻度でデータを取得しなければならない。表 3.4 の例では，サンプリング間隔が 60 秒未満のメトリックは３時間しか保存されないため，消去される前に管理サーバ側で取得しなければならない。そのため，AWS と管理ソフトにおける通信負荷が上昇することになる。なお，すべての監視メトリックがサンプリング間隔を調整可能であるとも限らない。AWS ではカスタムメトリックに出力されたデータのみ，60 秒未満の間隔で出力することができる。それ以外の出力を制御できるかどうかは，クラウドサービス側の実装に依存する。

**表 3.4 Cloudwatch のデータ保存期間**

| # | サンプリング間隔 | 保存期間 |
|---|---|---|
| 1 | １秒，５秒，10 秒，30 秒 | ３時間 |
| 2 | １分～４分 | 15 日間 |
| 3 | ５分～５９分 | 63 日間 |
| 4 | １時間～１日 | 455 日間 |

運用管理ソフトは，設定されたサンプリング間隔を単位とした統計値で稼働データを出力する。Cloudwatch ではサンプリング間隔ごとの統計値として合計値（SUM）・最大値（MAX）・最小値（MIN）を出力する機能を備えている。合計値（SUM）を観測サンプル要素の数で除算することで，平均値を取得することもできる。Cloudwatch は，サンプリング間隔の間に観測された稼働データを，その最初のポイント（時刻）の統計値として出力する。図 3.9 に示すように，12:00 から 12:04 までに内部では 1 分間隔でデータが観測されていても，サンプリング間隔が５分であれば，それらをすべて出力するのではなく 12:00 時点の値として統計値が出力される。AWS Cloudwatch の仕様ではその期間における最初のポイント（時刻）の統計値として出力されるが，その他の管理ソフトでは最終時刻の記録として出力されることもある。これらはソフトウェアやサービスの仕様に依存する。

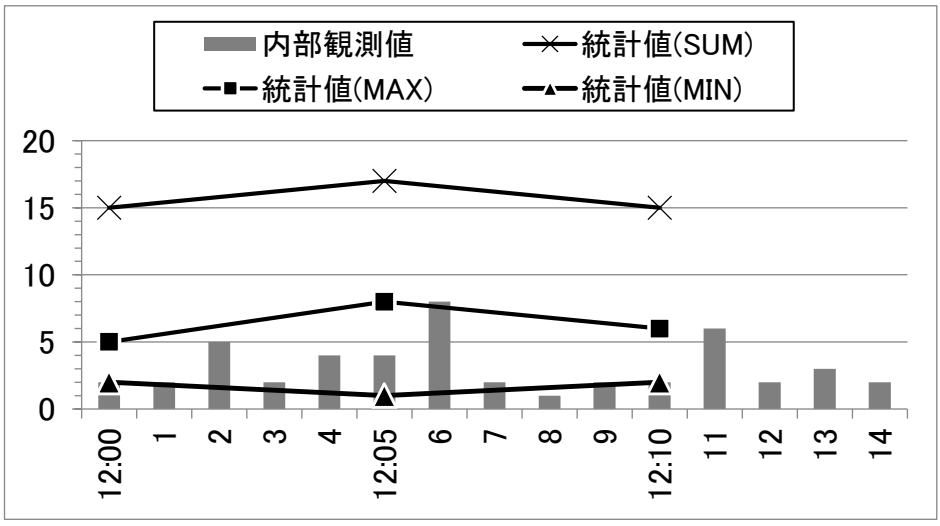



**図 3.9 運用管理データとして出力される統計値**

### 3.4.4. 運用管理チーム

運用管理プロセスを実行するためには，適切なチーム編成も重要である。本研究のスコープであるデータ保護については，従来はデータバックアップ専任組織を置くことが多かったが，現在ではいわゆる IT オペレーションチームがシステムの動作全体について責任を持つようになっている。サーバのプロビジョニングやデータへのアクセス，ソフトウェアの更新や稼働監視など多様なタスクを請け負う同チームが，データのトラブルに対しても復旧の責任を持つように運用管理の役割が変化している [50]。

## 3.5. デジタルデータの活用

### 3.5.1. データの変遷

グローバルに生成されるデータ量が爆発的に増加している [51] [52]。米 IDC 社の調査によれば，2016 年に 16ZB であった全世界のデータ量は，2025 年には 163ZB に達すると予想されている [2]。こうしたデータ増加に対応するために，データを収集・管理・処理および配送する技術がさらに進化する。その結果データ量が増え，それを処理する能力がまた向上するという好循環のサイクルが，今後のグローバルデータの爆発的な成長を牽引していく。

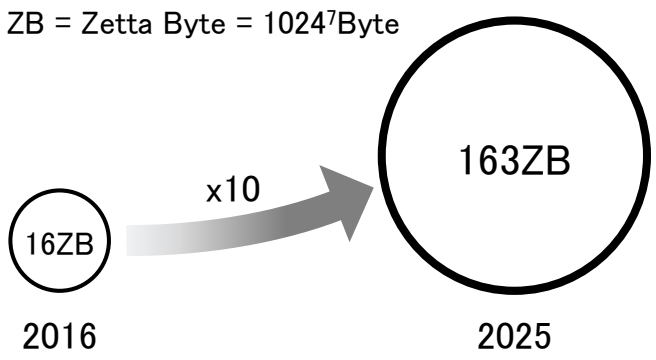



**図 3.10 世界のデータ量**

第一および第二のプラットフォームすなわち SoR で取り扱うデータは，リレーショナルデータベースのテーブル構造に定型化して格納された。特に金融取引をはじめとするトランザクション処理の記録や整理, 顧客情報の検索など, これらの定型データは「構造化データ」に分類される。

これに対して，画像や文書，電子メールなど多種多様なデータは「非構造化データ」に分類される。WEB サービスやモバイルサービス，SNS の普及に伴い，写真や動画をはじめとする非構造化データの生成量が飛躍的に増加した。非構造化データを取り扱うために旧来型のリレーショナルデータベースではなく, オブジェクトストレージなどの実装が求められる。これらの非構造化データを含めた大量データを示す概念は「ビッグデータ」と呼ばれる。ビッグデータを取り扱う並列分散処理技術や GPU などの技術発展が，データ分析を通じて新たな価値を探索するトレンドにつながっている [53]。

データの価値については，リアルタイムにアクセスできることへの可用性や，データが利用できなくなったときの重大さについての考察が必要である。たとえば医療アプリケーションが取り扱うデータは, エンターテイメント系の動画コンテンツよりもはるかに高い可用性が要求される。

PC の障害によって文書を消失することと，自動走行車両が人身事故を起こすこととは全く別の問題であり，後者はビジネスの存続を左右しかねるものである。このような hypercitical データの出現は，データのキャプチャ，分析および非常に高い信頼性・帯域

幅および可用性すなわちより安全なシステムを提供するインフラストラクチャの開発と展開を企業に求めるようになる [2]。

### 3.5.2. データ分析技術

こうした莫大な量のデータを対象とした分析により，新たな知見を得る技術が進歩している [54]。この分析にあたっては，データの種類に適合したコンピューティングプラットフォームを用意することが重要である。ここではデータの種類を“Data at rest”と“Data in motion”に分けてその違いを整理する。

**Data at rest**

様々なソースから収集され，記録・保管されている状態のデータ。記録から分析までに時間の経過があることを前提とする。大規模なデータからの学習が目的であるため，その“量”に重点が置かれる。

**Data in motion**

生成されたタイミングで処理するデータ。時系列で刻々と生成されるデータに対して，定常的に分析処理を適用する。リアルタイム処理を実行するため，その“速さ”に重点が置かれる。

“Data at rest”を処理するプラットフォームには，大容量のストレージで実装したデータレイクを設けることが前提となる。さらにそれらを並列計算するための計算機システムが必要となる。“Data in motion”にはリアルタイム処理を適用するため，低遅延処理に特化したシステムを設ける必要がある。すなわち記録を目的とした大容量データベースではなく，高速に流れるログに対して閾値検査するストリーミングデータベースや，特徴量を抽出する計算処理機能を備えた専用チップなどの実装が適している。

### 3.5.3. 人工知能

蓄積された膨大なデータから学習し，法則性やルールに基づくモデルを生成して特定

の判断を可能とする技術が機械学習である。WEB による大規模データと，クラウドによる大容量計算処理が可能となったことで，機械学習を適用した人工知能（Artificial Intelligence：AI）の研究開発が急速に進んでいる。高精度なモデルを構築することで，企業収益の拡大やコスト低減，より安全な社会の実現や自動走行車両などの新システムを実現することへの期待が高まっている [55]。

機械学習を実現するプラットフォームには Hadoop などの並列分散処理環境が適する。またモデリングを支援するライブラリである Tensorflow をはじめ，様々なソフトウェアが普及しはじめている。クラウドでも機械学習の実現に必要な分析・モデル作成・評価をはじめとする機能をサービスとして提供している。一例として，Microsoft Azure では機械学習の開発機能に加え，自動翻訳や動画の内容理解といった機能が利用可能である。

ニューラルネットワークを活用してデータから特徴を抽出する技術がディープラーニング（深層学習）である。パターン認識をするように設計された多層構造のアルゴリズムが DNN（Deep Neural Network）である。さらにその実装のひとつが DNN を 2 次元データに対応させた CNN（Convolutional Neural Network：畳み込みニューラルネットワーク）であり，画像認識に適している。さらに音声や動画など可変長のデータを扱えるように再帰構造をもたせたアルゴリズムが RNN（Recurrent Neural Network：再帰型ニューラルネットワーク）である。RNN は翻訳などの自然言語処理にも利用される。2016 年にはディープラーニングを用いた人工知能の囲碁プログラム「Alpha Go」がプロ棋士に勝利したことが話題となったが，その可能性を金融やヘルスケアなどビジネス分野に適用する動きが活発である。

### 3.6. 社会課題への挑戦

国家レベルでも，社会におけるあらゆる事象をデジタルデータに置き換えることで，様々な課題を解決する動きがはじまっている。内閣府は第 5 期科学技術基本計画において，日本が目指すべき未来社会の姿である Society 5.0 を提唱した [56] [57] [58]。Society 5.0 で実現する社会は，IoT で人とモノがつながり，様々な知識や情報を共有して新たな価値を生み出すことで，課題解決を目指していく。未来投資戦略 2017 では，Society 5.0

が目指す５つの戦略分野を宣言した。(1)健康寿命の延伸，(2)移動革命の実現，(3)サプライチェーンの次世代化，(4)快適なインフラ・まちづくり，(5)Fintech の５分野である。

その実現のキーとなるのがデータと人工知能（AI）である。Society 5.0 ではフィジカル空間のセンサから膨大なデータを収集し，サイバー空間に集積する。サイバー空間では，このデータを人工知能（AI）で解析し，フィジカル空間の人やモノにフィードバックする，いわゆるサイバー・フィジカル・システムによって，新たな価値を産業や社会に創出することを目指す。

### 3.7. 本研究の適用範囲

本研究では，情報システムが管理するデータを，適正なコストの範囲内で保護するための技術を提案する。本研究のアプローチでは，特に 3.2 節に述べたシステムアーキテクチャの種類には依存しない汎用的な方式を提案する。その検証にあたっては，ストレージサブシステムを活用したオープンシステムと，クラウドコンピューティング環境を対象として技術の試行と実験を行った。

また本研究では，性能シミュレーションによって適正なシステムサイズを導出する。3.3 節に述べたとおり，ウォーターフォール型開発ではシステム更改が容易ではなく，時間と工数もかかるため何度もやり直すことができない。そのためウォーターフォール型の現場にも適用できるよう，高精度なシミュレーション技術の確立を図る。また，負荷の増減に応じてシステムサイズを調整する方式を確立し，アジャイル型開発への適用も目指す。

本研究の性能評価方式およびシステムサイジング方式は，運用管理ソフトウェアに機能を拡張するアプローチを採用する。すなわち 3.4 節に述べた運用管理サブシステムに，運用管理データを入力とする予測計算機能を追加することで，リカバリポイントのシミュレーションを実現する。

3.5 節に述べた人工知能による価値の実現や，3.6 節に挙げた社会課題解決を牽引していくのはデジタルデータであり，その保護が今後より重要になっていく。次章ではデータ保護の重要性とその手段について考察する。

次に本研究のターゲットを時間軸で整理する。3.2.2 に述べたとおり，オープンシステムの世代にはそれぞれのシステムが個別の設計に基づいて開発された。その運用管理もまた，ソフトウェア環境や判定規則，閾値などのパラメータ設計を個別かつ綿密に設計することが求められた。さらに 2000 年代には，人手を介さずにシステムを運用する自己回復や自己最適化といった自律システム運用技術の研究が盛んに行われた。運用管理の自律化にあたっては，製品の振る舞いや過去の経験に基づいたベストプラクティスを If-then ルールで整備し，ナレッジベースに統合することで，システムパラメータの自動調整や障害回復の自動化が図られた。しかしながら個別のシステムごとに膨大な数の If-then ルールを実装することは困難であり，完全な自律化には至らなかった。

その後，クラウドが普及すると運用管理の手法も一変した。サーバ・ストレージ・ネットワークといったインフラは仮想化され，また画一的な仕組みで供給されるため，利用者側で詳細なシステム設計をする負担が大きく低下した。さらに稼働監視データもサービスとして提供されるようになり，運用管理システムも従来と比較すると簡易なもので足りるようになった。

一方，クラウドサービスはその仕様が非公開であることも多く，振る舞いを予測できないことも多い。また稼働監視も画一的であるため，必要なパラメータ（本研究のリカバリポイントや転送遅延時間）が標準サービスで提供されないこともある。そこで本研究では，クラウドで実装され，その振る舞いを直接コントロールできなくなった情報システムであっても適用できる，データ保護システム運用管理技術を提案する。

# 4章　データ保護の重要性と可用性向上に向けた取り組み

## 4.1. データ保護の重要性

3 章で述べたように，企業 IT システムはビジネスの遂行に必須の基盤である。それらは膨大な量のデータを創出し，データは参照できるよう管理される。また取引履歴などの業務記録やセンサが出力したログ，画像，地図などのビッグデータ分析や学習を通じて，新たな価値を発見しようとする試みが活発である。すなわち，ひとつひとつでは価値の無いデータであっても，膨大な量が蓄積されると，それ自体が価値を生む資産となる。こうした経営環境においては，企業が保管しているデータが消失すると，大きな損害が発生する。それは事業機会を失うだけでなく，顧客の信頼や社会的な信用を損なう可能性がある。状況によっては，企業の存続を左右する要因にもなり得る。

2012 年にはサーバホスティング事業者が，顧客から預かっていた 5700 契約分のデータをアップデート作業中に誤って削除してしまう事故が発生した。これらはバックアップを含めてすべて消失し，復旧もできなかった事案である。

2017 年にはソースコード管理サービスを提供する GitLab.com 社において，オペレータの操作ミスにより本番データベースのデータの大半を失う事故が発生した。このとき，同社が準備していた４つのバックアップ手段のうち，３つが機能していなかったことが判明した。結果的には偶然作成していた６時間前のスナップショットを使用して復旧に成功したが，障害発生時刻から遡って６時間前から同時刻までに更新されたデータは失われた。

企業では，様々なリスクへの対策をまとめた BCP（Business Continuity Plan : 事業継続計画）を準備するべきである [59]。BCP は自然災害，大火災，サイバーテロ攻撃などの緊急事態に遭遇したケースにおいて，事業資産の損害を最小限にとどめつつ，事業の継続あるいは早期復旧を可能とするための計画である。IT システムとデータを保護することは BCP で熟慮しておくべき項目のひとつであり，特にデータの重要性に応じた対策を準備しておくことが求められる。

## 4.2. データ保護計画

データ保護の計画にあたっては, 空間軸方向と時間軸方向の対策を考える必要がある [60] [61] [62] [63]。

(1) 空間軸方向への対策

単一拠点内でデータをバックアップするだけでは, 拠点規模の事故や大規模災害への対策にはならない。一例として東日本大震災規模の災害を想定した場合, 関東圏で運用するデータのバックアップを関西圏に保管するといった対策が求められる [64]。

本稿では, データを生成する拠点をローカルサイト, データの待避先となる拠点をリモートサイトと呼称する。想定する災害の規模や障害の波及範囲によって, これらのサイト間の距離を適切に設計しなければならない。データを冗長化して保護する際, 待避先であるリモートサイトへの地理的な距離が近すぎると, ひとつの災害でローカルサイトとリモートサイトの両方に問題が発生し, それぞれのデータを同時に消失する可能性がある。そのため待避先のサイトは想定される被災範囲の外に設置しなければならない [65]。

地理的な距離だけでなく, それぞれのサイトの電力系統を別にすることで, 同時停電のリスクを低減することも考えられる。また, 新興国では国境をまたぐ拠点を待避先とすることで, 政治的なリスクを抑えるほか, 有事の際に別系統の通信インフラでデータにアクセスできる可能性も高まる。

以上のようにデータ保全の観点からは, 待避サイトまでの距離を十分に空けることが望ましい。一方でサイト間の距離が遠いほどデータ転送には遅延や通信コストが生じる上に, 有事の際にリモートサイトで業務を復旧するための人員を配置することが難しくなる。そのためデータ保護の要件に応じて, 空間的な距離を適切に設計することが求められる [66]。

(2) 時間軸方向への対策

インシデントが発生した時刻を起点として, どの時点にさかのぼって復旧できるよう

にしておくか設計することも重要である。ここでは二つの考え方を整理する。ひとつはインシデントが発生した時点から，できるだけ新しいデータを復旧するための対策である。通常の定期バックアップでは，スケジュールされたタイミングで日次あるいは数時間おきにデータのスナップショットを取得する。この実行間隔を縮めることで，バックアップの取得時刻からインシデント発生時刻までの平均時間の短縮が期待できる。さらに，本番データの最終更新時刻とバックアップデータ取得時刻の時差を分単位あるいは秒単位に縮めるためには，定期バックアップではなく継続的にデータを複製する技術の適用が有効である。ストレージやデータベース，クラウドなどの機能を利用して，リモートサイトにデータを継続的に複製することができる。

もうひとつの時間軸への対策の考え方が，バックアップの世代数である。これは世代数を増やすことで，できるだけ古いデータを復旧できるようにするための施策である。日次のバックアップに加えて，過去のスナップショットを数日分あるいは数ヶ月分残すことで，ウィルスが混入する前のデータに復元する，あるいは監査対象となったある時点の取引履歴を再現するといった使い方ができる。

この両方の特徴をあわせもった解決策が CDP（Continuous Data Protection：継続データ保護）である。CDP は記憶装置へのデータの書き込みを常に監視し，その履歴を記録することで，任意の時点に復元可能とする方式である。最新の状態だけでなく，上書きされる前のデータも履歴に保存されているため，過去の時点にさかのぼって復元することを可能とする。データベースのジャーナル機能も CDP の実装手段のひとつである。一方，CDP は書き込み履歴をすべて記録するため，高い処理負荷が発生し，本番システムの性能に限界が生じる。また，履歴を記録するストレージの空き容量が枯渇すると，古い履歴から順に消去されるため，十分な時間を遡るためには，その履歴を記録する記憶容量が必要となる。こうしたデメリットもあるため，CDP は特にその適用が必要な場面に使用を制限するべきである。

以上の空間軸方向と時間軸方向への対策による解決策の分類を図 4.1 に示す。ローカルサイトとデータ待避先となるリモートサイトの距離は，大きく３段階に分類される。ローカルサイト内にバックアップを取得する構成，100km 未満の範囲内で離れたサイト

にデータを待避する構成, 100km 以上の距離を隔てたリモートサイトにデータを保管する構成の３つである。

縦軸はバックアップもしくはデータコピーの実行頻度による分類である。実行頻度が高いほど本番データの最終更新時刻とバックアップデータ作成時刻の差が短く, インシデント発生時に消失するデータ量を少なくすることを期待できる。

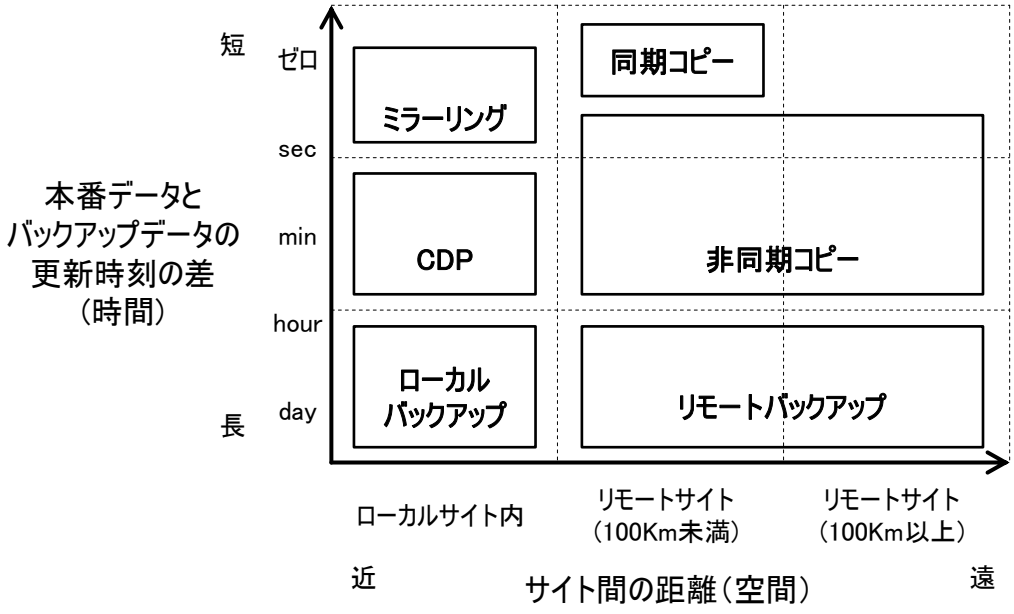


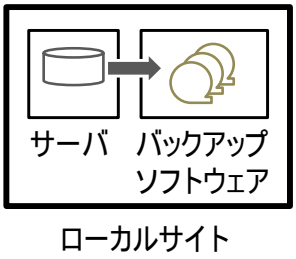
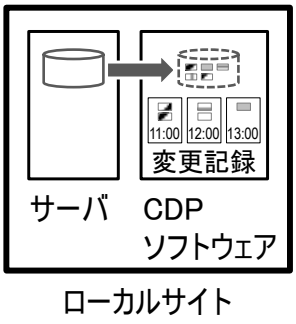
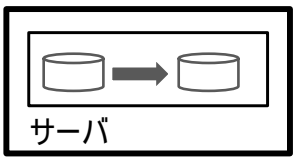

**図 4.1 データ保護方式**

これらの実行方式を表 4.1 および表 4.2 に示す。ローカルバックアップ方式 (#1) は, 夜間バッチ処理などスケジュールされたタイミングで, 同一サイト内のディスクやテープデバイスにバックアップを取得する。

CDP（#2）は前述のとおり，データ更新の履歴を記録することで，任意の時刻に復元するデータ保護方式である。これも同一サイト内に保護データを格納する実装が一般的である。

ミラーリング（#3）は，更新されたデータを単一メディアではなく複数の記録先に保存して冗長性を維持する方式である。２つのドライブに同一データを記録する RAID1 や，LVM（Logical Volume Manager）などソフトウェアにより実装できる。

表 4.1 データ保護方式（ローカルサイト内）

| # | 方式 | 概念図 | 空間軸 | 時間軸 |
|---|---|---|---|---|
| 1 | ローカル<br>バックアップ |  | サイト内 | Hour<br>Day |
| 2 | Continuous<br>Data Protection |  | サイト内 | Min<br>Sec |
| 3 | ミラーリング |  | サイト内 | Sec<br>遅延無し<br>(同期) |

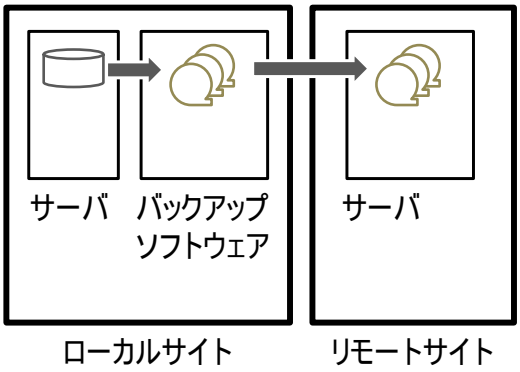
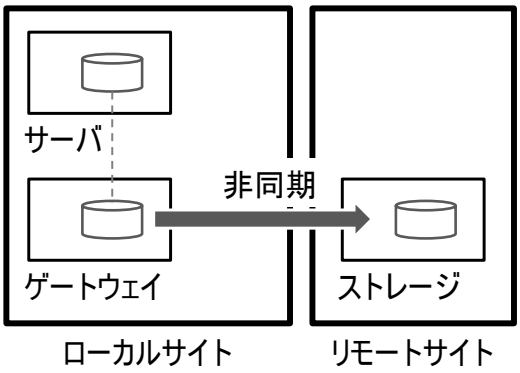
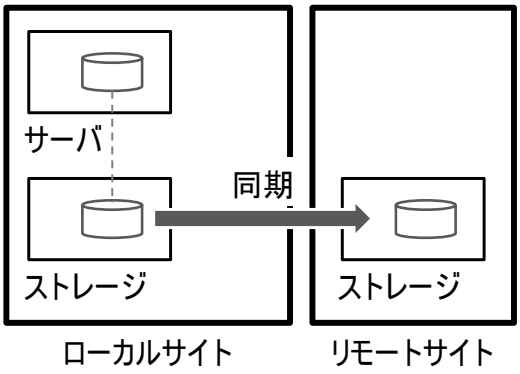
リモートバックアップ（#4）にもいくつかの実装形態があるが，一例としてローカルバックアップデータをさらにリモートサイトに複製する形態がある。ローカルサイトが被災しても，リモートサイトのバックアップソフトウェアによりデータを復元できる。その更新間隔はローカルバックアップのスケジュールに依存し，通常は日次もしくは数時間に一度といった頻度となる。

非同期コピー方式（#5）では，ローカルサイトで発生した更新データを継続的にリモートサイトへ転送する。高機能のストレージやデータベースは，更新データを即座にリモートへコピーする機能を備えており，その更新遅延を秒オーダより短くすることもできる [67]。

非同期コピーが分あるいは秒オーダの遅延を許容する方式であるのに対し，同期コピー（#6）は一切の遅延を発生させない方式である。サーバから書き込まれたデータがロ

ーカルサイトおよびリモートサイトに記録されたことを確認してからサーバに応答を返すため，それぞれのサイトに書き込まれたデータは完全に一致する。そのため時間軸方向のギャップは生じない。

**表 4.2 データ保護方式（ローカルサイト→リモートサイト）**

| # | 方式 | 概念図 | 空間軸 | 時間軸 |
|---|---|---|---|---|
| 4 | リモート<br>バックアップ |  | サイト間 | Hour<br>Day |
| 5 | 非同期コピー |  | サイト間 | Min・<br>Sec |
| 6 | 同期コピー |  | サイト間<br>(100km<br>未満) | 遅延無し<br>(同期) |

本研究は，ローカルサイトでのデータ保護ではなく，リモートサイトへのデータコピーによるデータ保護を対象とする。また，スケジュールされたタイミングで実行する定

期バックアップではなく，継続的にデータ転送する手法に着目した技術を提案する。この手法には同期コピーと非同期コピーがあるが，このうち非同期コピー方式（#5）が本研究の適用領域に該当する。

### 4.3. データ消失のリスクコントロール

時間軸方向のデータ保護性能を表す指標には，RPO（Recovery Point Objective）と RTO（Recovery Time Objective）の二つが用いられる [68]。RPO は「リカバリポイント」の目標値である。リカバリポイントとは，本番データに問題が生じた「インシデント発生時刻」と，同時刻からさかのぼって回復可能であるデータの更新時刻の時差である。例えば，RPO が 24 時間と定義されていれば，インシデント発生時刻からさかのぼって 24 時間前の過去データに復旧できるよう，データ保護システムを計画することが求められる。これは 24 時間前に作成・更新されたデータの保護・復旧を常に保証する要件定義に他ならない。言い換えれば，RPO が 24 時間であるということは，インシデント発生からさかのぼって 24 時間以内に更新されたデータの消失を許容することを意味する。データならびに業務の重要度によって，データ消失のリスクをどの程度許容するかを定め，RPO を設計することが必要となる [25] [69]。

もうひとつの性能指標である RTO はインシデント発生から復旧までにかかる時間の目標値である。

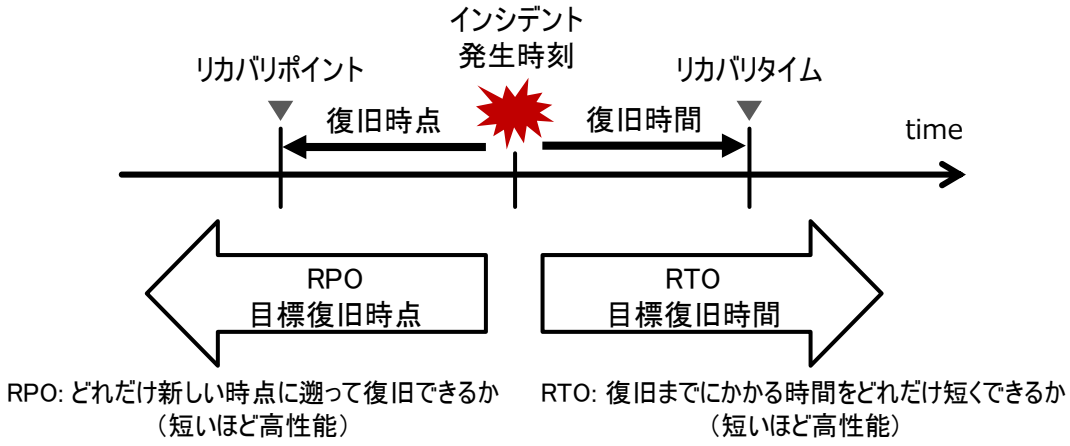



**図 4.2 データ保護性能指標**

RPO と RTO はどちらもゼロに近い方が事業継続性は高まるが，その分システムの構築コスト（CAPEX：Capital Expenditure）や運用コスト（OPEX：Operating Expense）が大きくなる。IT システムにかかる費用は，企業経営にとって利益を圧迫する要因のひとつである。そのため，RPO はデータ消失リスク，RTO はシステムダウンタイムを許容するように，経営判断を伴って計画することが望ましい。

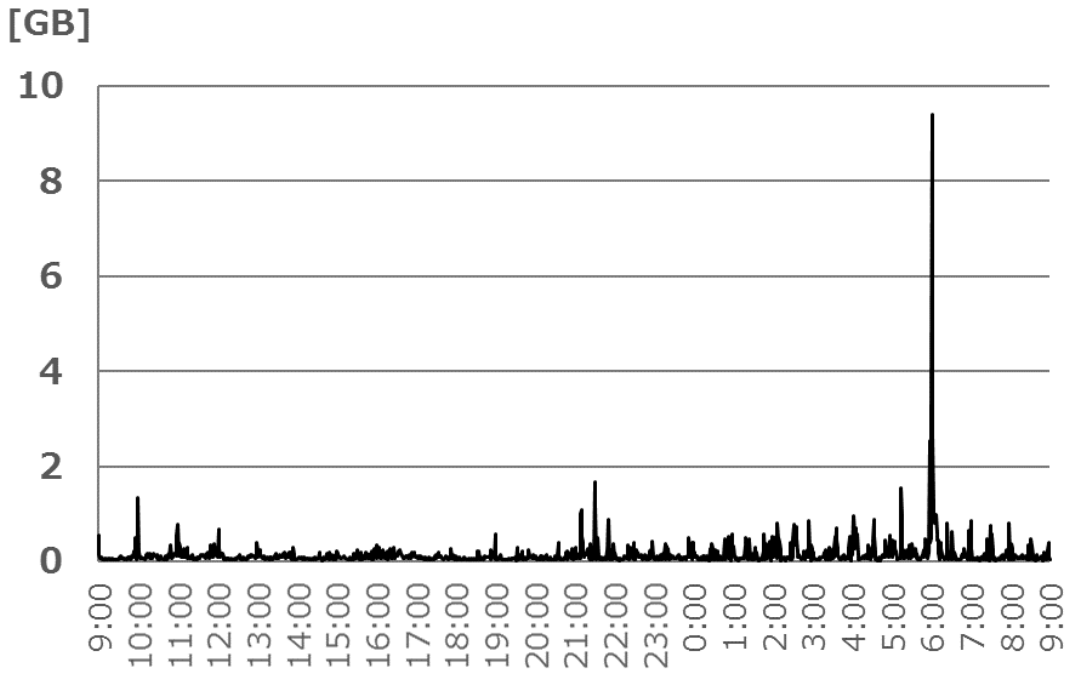



**図 4.3 実測したデータ書き込み量の例**

システム設計の段階では，その能力を計画する「キャパシティプランニング」と呼ばれる業務を遂行する。高信頼性と安定稼働が求められる企業 IT システムでは，負荷のピークにあわせたシステム能力（キャパシティ）を設けることが一般的である。しかしながら，突発的な負荷のピークにあわせたシステム構成では，通常ほとんど利用されない資源が発生する可能性がある。実際の企業システムで発生したストレージへの書き込みデータ量の推移例を図 4.3 に示す。この例では，突発的な負荷である 9.6GB/min が 6:00 に発生しているが，それ以外の時間帯では低負荷状態が続いている。この場合，ピ

ーク負荷にあわせて設計すると，6:00 を除く時間帯では使用されない冗長リソースが実装される。もしも，このシステムがピーク負荷の時間帯における一時的な性能低下を許容できれば，冗長なシステム資源の実装によるコストを削減できる可能性がある。

データ保護システムの設計においては，高負荷時のデータ消失リスクをコントロールしながら，全体のコストパフォーマンスを向上させることが重要な要件となる。またその設計においては，性能を高精度に見積もることが求められる。

### 4.4. データ保護システム

システムの継続性やデータの重要性に基づいて，データ保護システムの実装方式を適切に選択する必要がある。過去にはバックアップデータを記録したテープ媒体をトラックで遠隔地に搬送し，保管するといった災害対策も実施されてきた。しかしながらこの方式では復旧に一日以上の時間を要し，ダウンタイムが大きな損失につながる業務には適さない。データをネットワークで遠隔地に転送するデータ保護システムの実装形態の例を表 4.3 に示す。

**表 4.3 データ保護システムの実装形態**

| # | 実装形態 | 対象データ | 更新の速さ | 更新タイミング |
|---|---|---|---|---|
| 1 | バックアップソフトウェア | OS，テーブル，仮想サーバ他 | 低 | スケジュールされたバックアップ取得のタイミングで転送 |
| 2 | データベース | テーブル | 高 | トランザクションログ発生のタイミングで即時転送 |
| 3 | サーバ仮想化ソフトウェア | 仮想サーバ，仮想ストレージ | 中 | 設定値（最短５分，最長 24 時間）のタイミングで転送 |
| 4 | ストレージシステム | ボリューム，ファイル | 高 | 書き込まれたデータを即時転送 |
| 5 | クラウドサービス | ボリューム，ファイル，テープ | 中 | 分単位でバッファに一時待機した後，適時転送 |

### 4.4.1. バックアップソフトウェアによる実装

企業データセンタ向けのバックアップソフトウェアには, 遠隔地へのデータ複製機能を備えたものもある [70]。一例として, 米 Veritas Technologies 社の製品である NetBackup は，様々な OS，データベース，サーバ仮想化ソフトウェア，ストレージシステムならびにクラウドのデータ保護機能との連携をサポートする, 統合バックアッププラットフォームである。様々なデータソースを対象とするバックアップを統合運用する。同ソフトウェア自体がデータレプリケーション機能（Auto Image Replication）を有し，これらのバックアップイメージを遠隔地へ複製する（図 4.4)。このとき，転送データ量および保存するデータ量を削減するために，ローカルサイト側で重複排除を適用し，専用ディスクに格納する。また，データベースやストレージが提供するレプリケーション機能と連動する実装もできる他, クラウドストレージをバックアップデータの保存先に指定する構成も可能である [22]。これらのバックアップ専用システムの構築ならびにスケジューリングなどの運用管理が必要であるが, 汎用的なデータ保護の解決策として広く利用されている。

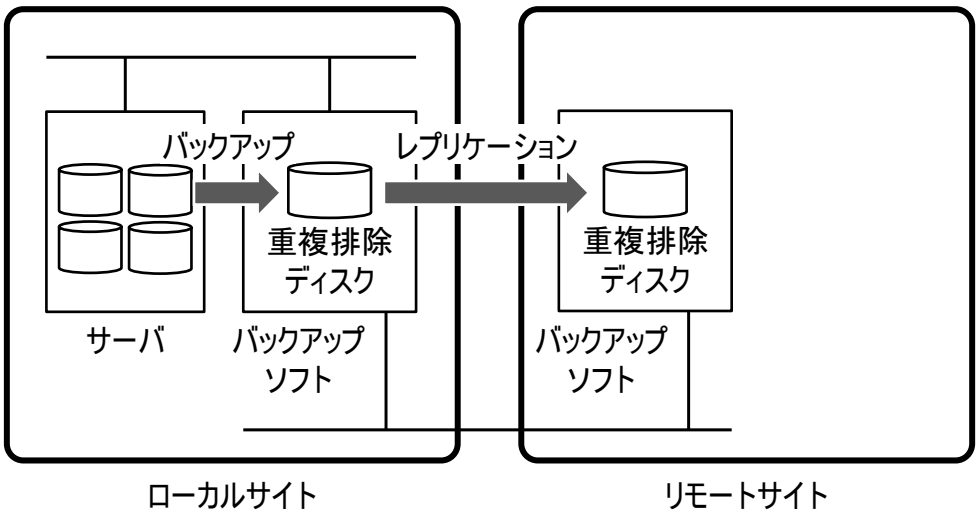



**図 4.4 バックアップソフトウェアによる実装**

### 4.4.2. データベースによる実装

データベースの保護にあたっては，レプリケーション機能を活用できる。企業向けSoRではOracleデータベースが提供するData Guard [71]が普及している他，Mongo DBやPostgreSQLもレプリケーション機能を備えている。Oracle Data Guardは指定されたテーブルを対象として，更新されたトランザクションログを遠隔地に転送する。その転送モードには同期ログ転送と非同期ログ転送がある。同期転送では，プライマリデータベースに発生したログを転送し，スタンバイデータベースの受信確認を待ってからアプリケーションにコミットの応答を返す。このため同期転送ではプライマリとスタンバイデータベースが常に一致し，データ消失が発生しないことが保証される。これに対して非同期転送では，プライマリデータベース側でログを記録すると，スタンバイデータベースでの受信確認を待たずにアプリケーションへ応答を返す。長距離データ転送遅延に起因する性能低下を回避できる一方，未転送状態のデータを消失するリスクがある。

データベースによるレプリケーション機能を利用すれば，テーブル単位でデータ保護システムを設計できるメリットがある一方，データベースサーバでレプリケーション処理を実行するための負荷が発生し，本番のトランザクション処理性能を低下させるリスクがある。

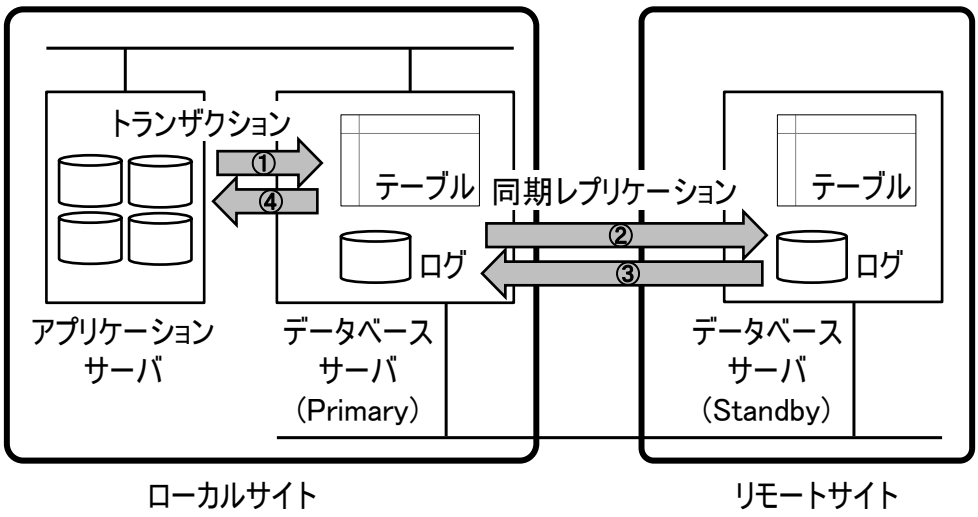



**図 4.5 データベースによる実装（同期転送方式）**

① トランザクション処理の書き込みを要求

② トランザクションログをスタンバイデータベースにコピー

③ スタンバイデータベースから受領確認を通知

④ アプリケーションへコミットの応答を通知

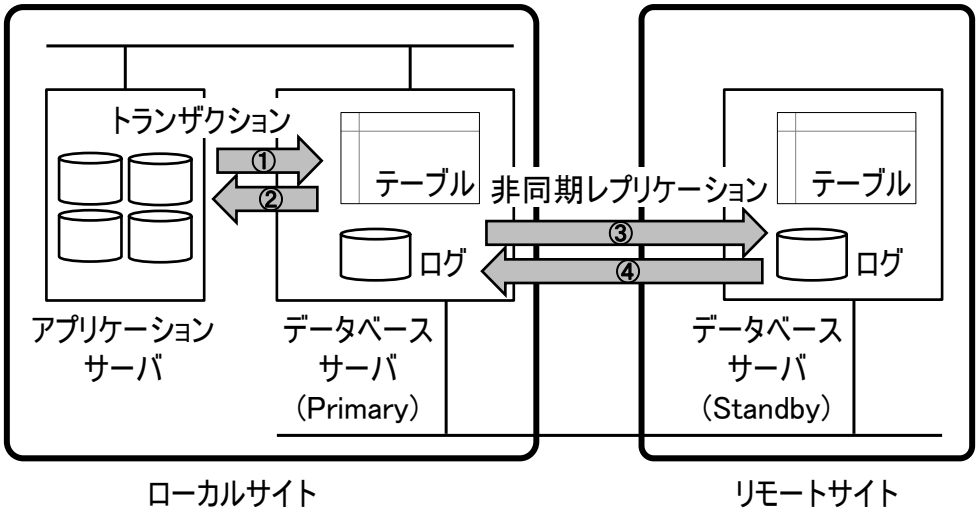



**図 4.6 データベースによる実装（非同期転送方式）**

① トランザクション処理の書き込みを要求

② アプリケーションへコミットの応答を通知

③ トランザクションログをスタンバイデータベースにコピー

④ スタンバイデータベースから受領確認を通知

### 4.4.3. サーバ仮想化ソフトウェアによる実装

計算機やストレージ資源を仮想化するサーバ仮想化ソフトウェアもデータを遠隔地へ複製する機能を備えている。VMware 社が提供する vSphere Replication は，仮想サーバと，その記憶領域である仮想ストレージを対象としたデータ複製機能である。物理サーバで稼働するハイパーバイザが，リモートサイトのハイパーバイザと連動して指定された仮想サーバを複製する。操作対象が仮想化されたリソースであるため，リモートサイトの物理サーバならびに物理ストレージは，ローカルサイトとは別の種類のものであ

っても構わない。ローカルおよびリモートサイトを，同環境向けの管理ソフト vCenter で統合運用できる点も特徴である。

vSphere Replication では，RPO を５分から 24 時間の間で設定することができる。これは，更新されたデータを最短５分以内にリモートサイトへ複製することを意味している。逆に言えば，RPO を５分より短く設定したい IT システムには適用できないという制限でもある。

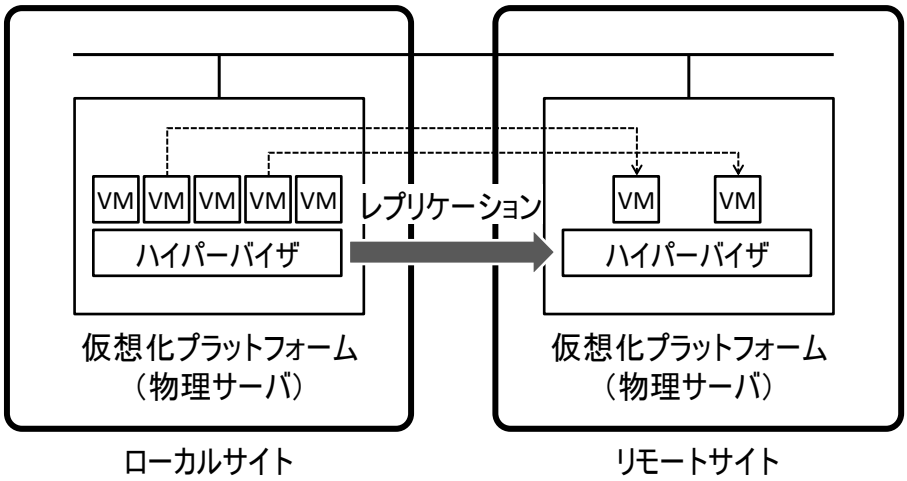



**図 4.7 サーバ仮想化ソフトウェアによる実装**

### 4.4.4. ストレージシステムによる実装

特に短時間の PRO を実現するための技術のひとつが，ストレージシステムによるリモートコピー機能である。リモートコピー機能は，ストレージが供給する記憶領域であるボリュームを対象として，同ボリュームに書き込まれたデータを順次，対となるボリュームに複製する。リモートコピー機能には同期方式と非同期方式がある [67] [68]。

同期コピー方式では，ストレージが提供するデータボリューム（正ボリューム）に書き込まれたデータをキャッシュに記録し，ブロック単位でリモートサイトにあるストレージに転送する。リモートサイトのストレージでは，ペアになるボリューム（副ボリューム）への書き込みが完了すると応答を返し，その受信を確認してからサーバに書き込み完了通知を返す。このため同期コピー方式では正ボリュームと副ボリュームの内容が

常に一致し，データを消失しないことが保証される。この方式ではデータの書き込み開始から完了報告を受信するまでに，サイト間の距離に起因する転送遅延が発生し，サーバへの応答性能が低下する可能性がある。そのため，同期コピーを適用する距離は100Km 以下とすることが推奨されている。

これに対して非同期コピーではストレージが書き込み要求を受領すると，すぐに書き込み完了通知をサーバに返す。複製するデータはいったんバッファであるジャーナル領域に格納された後，FIFO 処理にしたがってリモートサイトストレージに転送する。同方式では，アプリケーションサーバが書き込み完了を確認したステータスであっても，リモートサイトへの転送を終えていないデータがジャーナル領域に残存する可能性がある。長距離データ転送遅延に起因する性能低下を回避できる一方，ローカルサイトが被災した場合にジャーナルに残存するデータを消失するリスクがある [72] [73]。

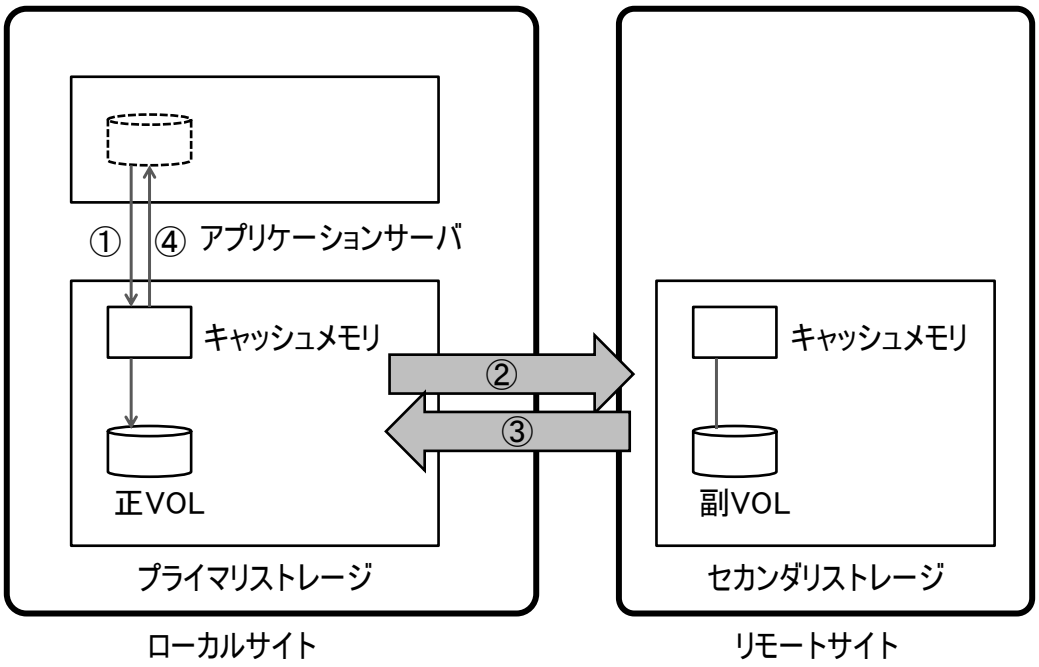



**図 4.8 ストレージによる実装（同期転送方式）**

① サーバからのデータ書き込み要求

② セカンダリストレージへのデータ転送

③　セカンダリストレージから書き込み完了を通知

④　サーバに書き込み完了の応答を通知

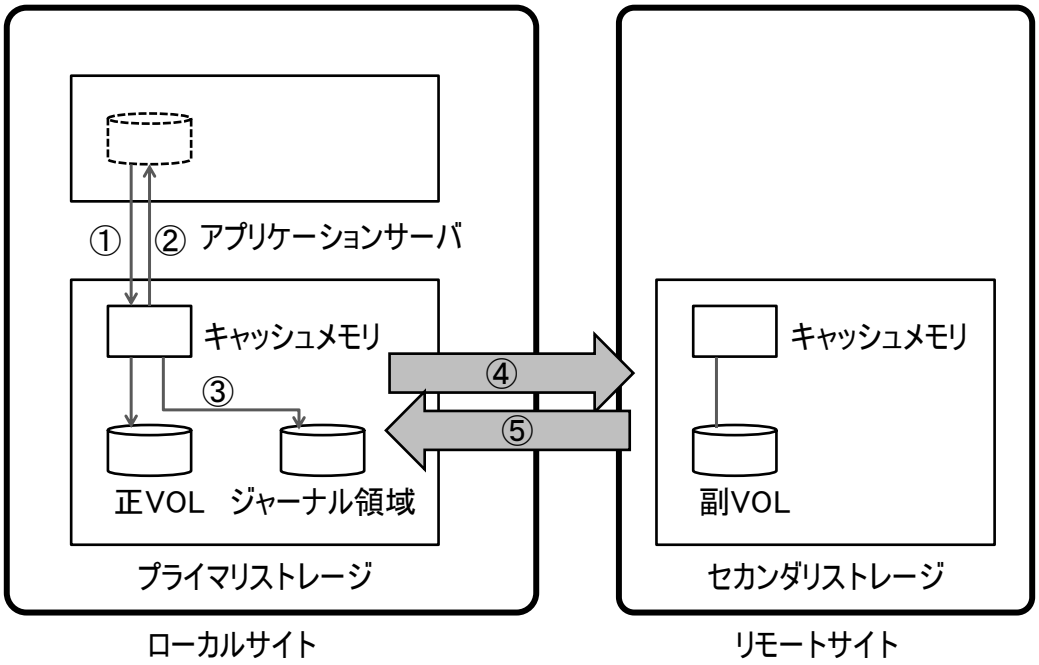



**図 4.9 ストレージによる実装（非同期転送方式）**

①　サーバからのデータ書き込み要求

②　サーバに書き込み完了の応答を通知

③　書き込みデータをジャーナル領域に格納

④　セカンダリストレージへのデータ転送

⑤　セカンダリストレージから書き込み完了を通知

### 4.4.5. クラウドサービスによる実装

クラウドサービスもまた, データの退避先として有効なソリューションのひとつである。クラウドを利用することで, バックアップデータを保管するデータセンタを企業が自ら運用するコストを排除することができる。一般的にクラウド事業者は, 複数の拠点にまたがってデータを二重化あるいは三重化する機構を備えることで可用性・信頼性を

高めており，データの保管先としても適切であるといえる [74] [75] [76]。

Amazon Web Services や Microsoft Azure といったメジャーなクラウド事業者では，企業がオンプレミスに設置したデータセンタで発生したデータを，クラウドへ待避するためのサービスを提供している。AWS Storage Gateway は，オンプレミス環境とクラウドストレージを接続するゲートウェイを提供するサービスである [77]。同ストレージゲートウェイは，オンプレミスで作成されたデータを，順次クラウドストレージに転送する。オンプレミス環境に対しては，NFS など業界標準のファイルインタフェースを提供するファイルゲートウェイ，iSCSI デバイスとして動作するボリュームゲートウェイ，仮想テープライブラリとしてテープへの書き出しをクラウドに記録するテープゲートウェイの３つのモードからいずれかの動作を選択する。このゲートウェイの実装は，オンプレミス環境に仮想マシンとしてインストールする他，あるいは AWS の仮想サーバ（EC2 インスタンス）としてインストールする形態を選ぶことができる。

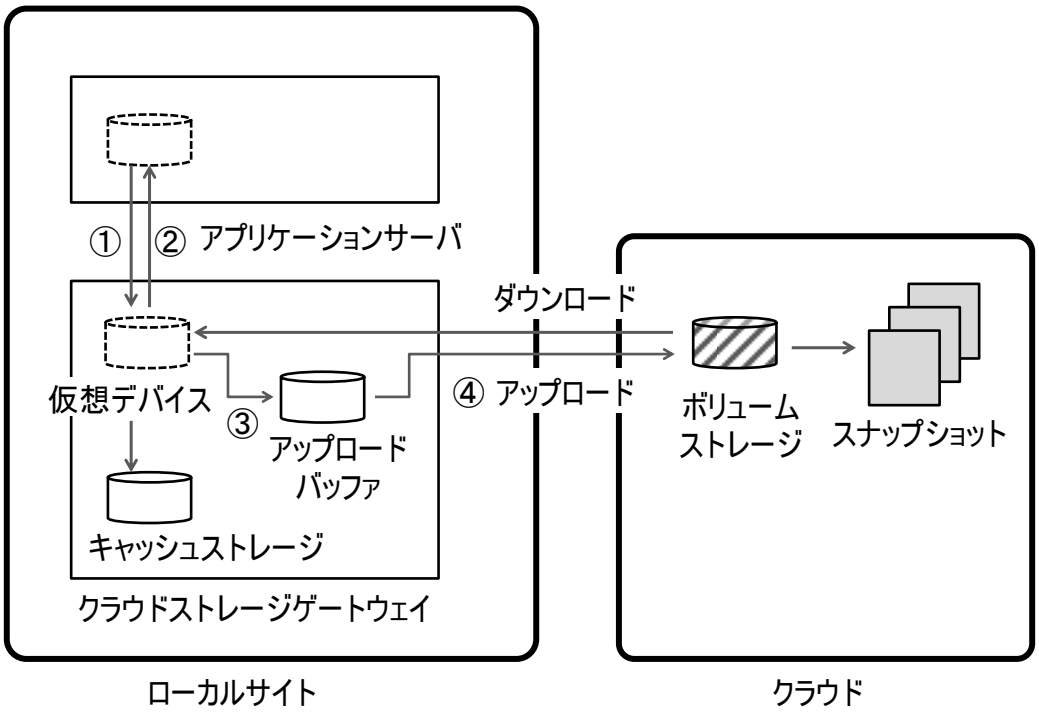



**図 4.10 クラウドゲートウェイによる実装（キャッシュ型ボリューム）**

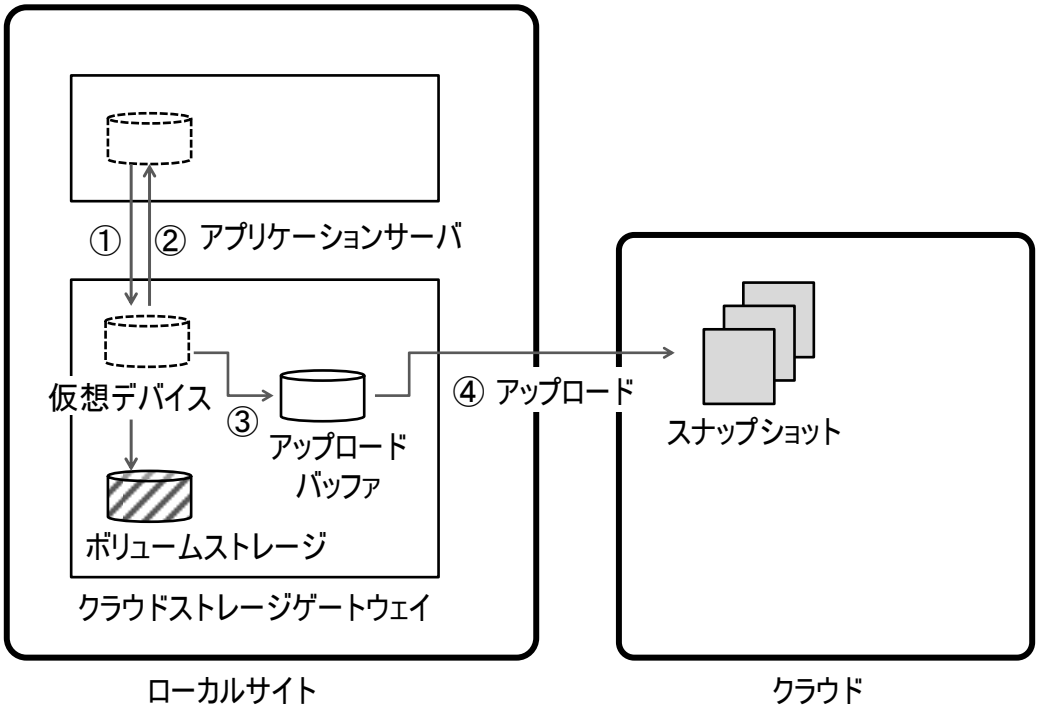



**図 4.11 クラウドゲートウェイによる実装（保管型ボリューム）**

このストレージゲートウェイをオンプレミス環境に設置した場合の構成を図 4.10, 図 4.11 に示す。ここでは本研究の対象であるボリュームゲートウェイのアーキテクチャを例示している。ボリュームゲートウェイには，プライマリデータをクラウドに置きゲートウェイをキャッシュとして用いる「キャッシュ型ボリューム」と，プライマリデータをゲートウェイに置き，非同期でクラウドにデータを転送する「保管型ボリューム」の二種類の動作モードがある。

いずれの動作モードでも，ストレージゲートウェイはオンプレミスのアプリケーションサーバに対して仮想デバイスを提供する。アプリケーションサーバは iSCSI プロトコルでこれらを認識し，保護するデータの格納場所として使用する。

キャッシュ型ボリュームでは書き込まれたデータをキャッシュストレージに保存する。キャッシュストレージにはローカルのストレージ環境（SAN, NAS など）を利用し，

アプリケーションサーバの入出力で頻繁にアクセスされるデータに低遅延でアクセスするためのキャッシュデータを格納する。書き込まれたデータはクラウドへ転送する前にアップロードバッファにいったん待避する。ストレージゲートウェイはここからデータを暗号化してクラウドへアップロードする。

保管型ボリュームでは，記録されたすべてのデータをオンプレミスの記憶領域であるボリュームストレージに保持する。更新されたデータのみ，スナップショットを取得するタイミングでクラウドへ転送する。転送する前にはアップロードバッファにいったん待避し，ここから暗号化してクラウドへ送られる。

### 4.5. 本研究の狙い

4.1 節で述べたとおり，データに関わる障害は企業に重大な不利益を生じる可能性がある。そのため 4.2 節に述べたように，データ保護システムの要件を正しく設計し，特に RPO をはじめとする性能要件を達成するように運用することが重要である。GitLab.com 社の事例では，24 時間ごとのバックアップを計画していたので，RPO は 24 時間であったと推測される。この事例では，結果的に障害発生時刻からさかのぼって 6 時間前のデータに復旧できたことから，RPO は達成されたことになる。すなわち障害発生前 6 時間の間に更新されたデータは失われたが，これは設計段階で定義したリスク許容範囲内であったと解釈できる。このようにデータ保護システムの運用においては，RPO を達成していることかどうか定常的に監視する仕組みが求められる [17]。

定期バックアップの成否だけでなく，4.4 節に述べたように継続的にデータ転送するシステムにおいて，この RPO は特に重要である。データベースやストレージシステム，クラウドサービスの非同期コピー機能は，秒オーダあるいは分オーダといった短時間の PRO を達成するために用いるものだが，その運用においてリカバリポイントを定常的に監視する手段が必ずしも提供されないという問題がある。そこで本研究では，稼働しているデータ保護システムが RPO を達成しているかどうか，特別な機能実装をせずに判定するための技術を提案する。

さらに 4.3 節に述べたように，データ保護システムはできるだけコストを抑えることが望ましい。そこで本研究では，ある想定のシステム構成を対象として，性能すなわち

リカバリポイントをシミュレーションにより予測する技術を提案する。同技術を使用することで，性能（リカバリポイント）がその目標値である RPO を達成し，かつリソース量を削減することでコストを抑えたシステム構成を導出することを目論む。

すなわち，本研究が確立しようとする技術は以下の二つである。

(1) データ保護システムの性能であるリカバリポイント評価技術

(2) 最適なシステムサイズを導出するための性能シミュレーション技術


**図 4.12 データ保護システム運用管理の PDCA**

これらは，データ保護システムの運用管理における PDCA サイクルの一部を実現する技術である。図 4.12 に示すとおり，3.4 節に述べた運用管理の PDCA をデータ保護システムに適用すると，まず Plan フェーズでは RPO や RTO の目標値を適切に計画することが求められる。経営判断にもとづき，データ消失リスクをどの程度許容するべきか，すなわち RPO を適切な値に定義する。Do フェーズでは，実装したデータ保護システムのリカバリポイントを監視し，RPO を達成しているかどうかを検査しながらシステムを運用する。Check フェーズでは，実際に発生した負荷とリカバリポイントを調査し，データ保護性能が適正か，あるいは過剰な性能とコストになっていないかどうかを検証

する。そして Action フェーズでは，期待される性能を発揮するために必要なシステムリソース量を算出し，キャパシティプランニングを遂行する。

このうち，(1)データ保護性能評価技術は Do フェーズと Check フェーズを，(2)データ保護性能シミュレーション技術は Check フェーズから Action フェーズを実行するための技術に位置づけられる。

# 5章　データ保護システムの性能評価技術の提案

## 5.1. データ保護システム性能評価における課題

非同期コピーを適用したデータ保護システムは，転送データをバッファに一時的に保管する。このとき災害などのインシデントが発生すると，バッファに蓄積しているデータは消失する。その時点のリカバリポイントは，RPO より短時間であることが求められる。

図 5.1 はあるデータ保護システムの 12 時 05 分時点のデータ転送の様子を表現している。データに付与されたタイムスタンプは，そのデータの生成時刻を示す。12 時 00 分にサーバで生成されたデータが，バッファでの待機や長距離通信による遅延を経て 12 時 05 分にリモートサイトへ記録されたことを表している。この時点でローカルサイトに障害が起こると，バッファに残っている 12 時 00 分より後に生成されたデータは失われる。このときのリカバリポイントは，インシデント発生時刻である 12 時 05 分とデータ復旧可能時刻 12 時 00 分の差である「５分」に一致する。

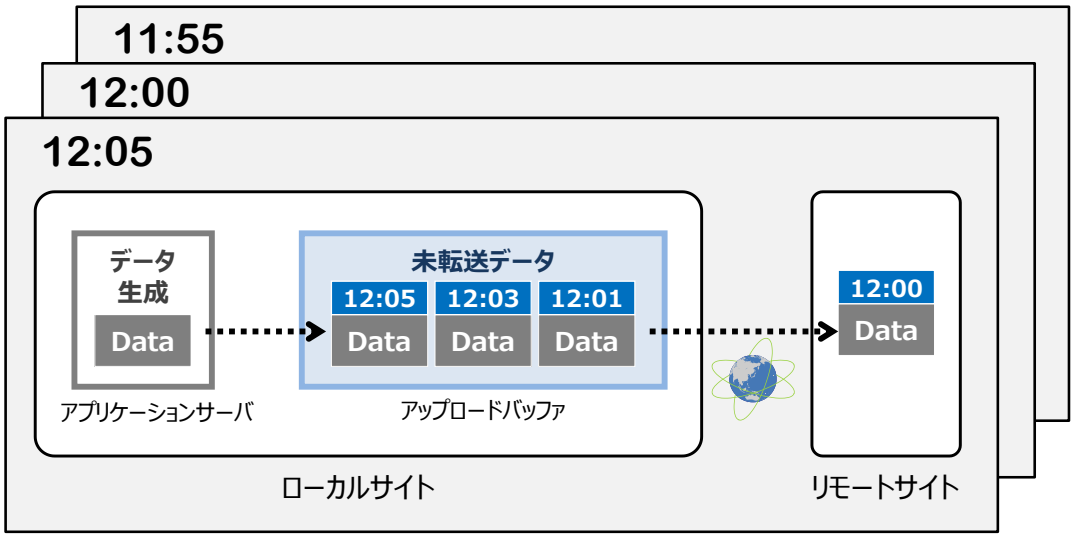



**図 5.1 未転送データとリカバリポイント**

データ消失リスクを管理するためには, 時系列で刻々と変化するリカバリポイントを監視し，RPO を達成しているかどうか判定しなければならない。リカバリポイントが RPO より短時間であれば, データ保護システムは性能要件を満たしており，リスクも許容できる範囲内である。もしリカバリポイントが RPO より長時間になる状況が発生した場合, 管理ソフトウェアから管理者にアラートを通知して, 対策することが望ましい。しかしながら，一般のシステム管理ソフトやクラウドのシステム監視サービスは，リカバリポイントに相当する監視メトリックを提供しない。一例として, AWS Cloudwatch では，リカバリポイントに該当するメトリックは出力されない。データベースやストレージシステムの運用管理ソフトでも同様である。その理由のひとつが，図 5.1 に示すようなタイムスタンプの処理が実装されないことである。膨大な量のデータ転送によって発生する計算処理のオーバヘッドを回避するため, こうしたタイムスタンプ処理は通常のデータ転送システムでは実装されない。

## 5.2. 課題の解決方針

本研究では，タイムスタンプなどの特別な機能実装を必要としなくても，標準的なサービスでサポートされている他のメトリックを入力として, リカバリポイントを計算によって導出することを目指す。図 5.2 に示すように，本技術は運用管理サブシステム側の工夫だけで課題解決を図る。ベンダあるいはクラウドサービスプロバイダから提供されるデータ保護システムの本番系の処理に影響を与えることなく, 運用管理データを参照する機能追加だけで実現できる汎用的な手法の確立を目指す。

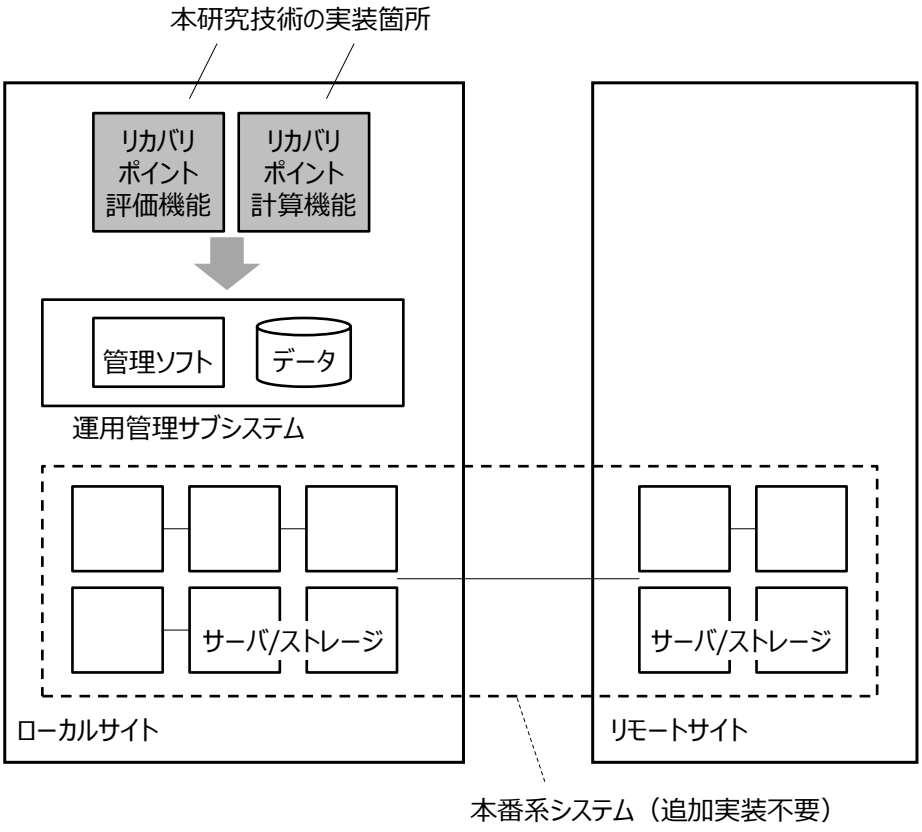



**図 5.2 本研究技術の実装箇所**

また本研究では，4.4 節に挙げた様々なシステム実装に適用できる汎用的な計算方法の確立を目指すため，ある実装に特化するのではなく，データ保護システムの構成や振る舞いを抽象化したモデルで表現する。このモデルに従ってリカバリポイントを評価可能とする計算式を考案する。

リカバリポイントの計算にあたっては，(1)サーバからの書き込みデータ量，(2)バッファに滞留する未転送データ量の時系列推移，ならびに(3)データ転送性能を入力とした計算手順を策定する。(1)と(2)は一般のストレージ管理ソフトやクラウド監視サービスでもサポートされている標準的なメトリックであり，容易に利用できる。(3)はシステム設計段階で既知の定数であり，またシステムの稼働監視結果から性能の上限値を推測することもできる。これらの有効なパラメータを入力としたリカバリポイント計算式の入出力の関係を図 5.3 に示す。このリカバリポイント計算結果と RPO を比較し，性能

目標値である RPO を達成しているかどうか判定する。

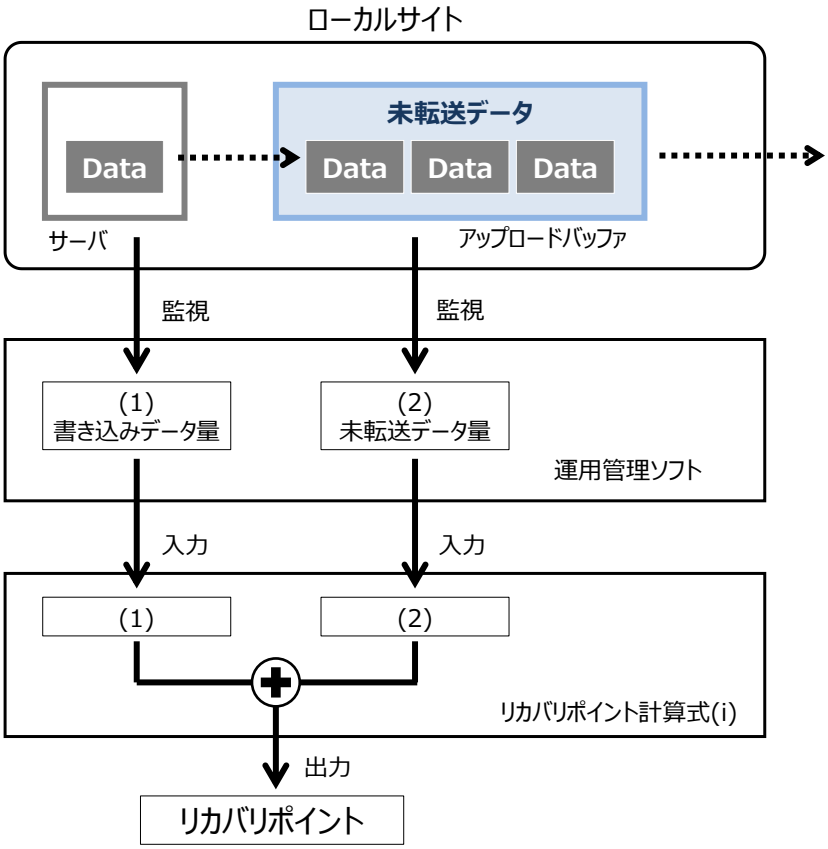



**図 5.3 リカバリポイント計算の入出力フロー**

## 5.3. データ保護システム性能評価方式

### 5.3.1. システムモデリング

リカバリポイントを計算可能とするために，まずデータ保護システムの構造ならびに振る舞いをモデル化する。表 4.3 に述べたとおり，データ保護システムの実装形態は様々であるが，本研究は非同期コピー方式を対象としたリカバリポイント計算について考察する。非同期コピーを適用したデータ保護システムのモデルを図 5.4 に示す。

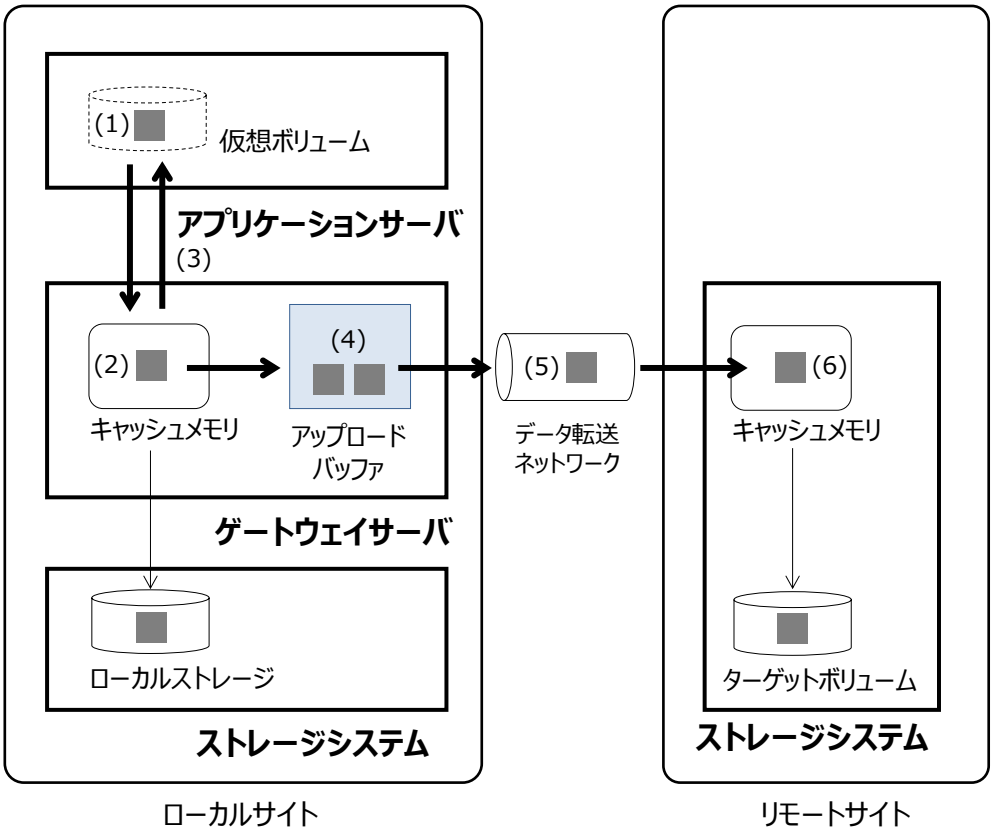



**図 5.4 データ保護システムモデル**

このデータ転送システムのアーキテクチャでは，アプリケーションサーバ，ストレージシステム，ゲートウェイサーバをローカルサイトに設置し，データ転送ネットワークを介してデータの転送先であるリモートサイトのストレージシステムに接続された構成とする。ゲートウェイサーバはデータを格納する記憶領域である仮想ボリュームをアプリケーションサーバに提供する。アプリケーションサーバは SCSI もしくは iSCSI などの標準プロトコルで仮想ボリュームを認識し，ファイルシステムにマウントする。ストレージシステムは仮想ボリュームの記憶容量をゲートウェイに提供する。さらにゲートウェイには，リモートサイトへ送出する前に一時的に転送データを保管するアップロードバッファ領域を設ける。リモートサイトはバックアップデータの格納先となるターゲットボリュームを提供する。

このモデルのシステムは以下のように振る舞う。まずアプリケーションサーバが，生成したデータを仮想ボリュームに書き込む（処理１）。ゲートウェイではこれをキャッシュメモリに記録し（処理２），サーバに書き込み完了通知を返す（処理３）。次に転送するデータをアップロードバッファに格納する（処理４）。ゲートウェイサーバは書き込まれた順序を変更せず FIFO 処理で，データ転送ネットワークを経由してバッファに滞留するデータをリモートサイトへ転送し（処理５），リモートサイトではこれをストレージシステムに記録する（処理６）。同モデルによれば，処理３でサーバに完了通知を返した時刻が，各データの書き込み時刻に相当する。また，処理６でリモートサイトのストレージに記録された時刻が，同データの転送処理が完了した時刻である。したがって，処理３で記録されたデータは，処理６で転送完了するまで保護されておらず，インシデントの発生によって消失する状態にある。なお，キャッシュに記録されたデータをローカルストレージならびにターゲットボリュームへ書き出す「デステージ処理」は，データ転送とは別のプロセスで実行するため，一連の処理の流れからは除外している。

これらの一連の処理フローのうち，処理４におけるアップロードバッファのデータ入出力性能と，処理５におけるデータ転送ネットワークの通信性能が全体のデータ転送処理性能の上限になり得る。書き込まれたデータ量，すなわち書き込み負荷に対してこれらの性能が不足すると，データ転送処理が滞り，アップロードバッファに未転送データが蓄積する。したがってこれらの未転送データを格納できるだけのストレージ容量をアップロードバッファに設ける必要がある。以上の条件により，同モデルの適用にあたっては，(a)データ転送ネットワークの通信性能（通信帯域），(b)アップロードバッファの入出力性能に加え，(c)アップロードバッファの容量を変数としたシステムリソース設計が求められる。

同モデルは，図 4.9 のストレージシステムを抽象化した構造を表現している。図 5.5 ではストレージが同モデルのゲートウェイサーバ各機能を包含した構成となっており，正ボリュームがローカルストレージに，ジャーナル領域がアップロードバッファに相当する。副ボリュームはリモートサイトに設置されたストレージのターゲットボリューム

に該当する。

同時に，同モデルは図 4.10 のクラウドゲートウェイを抽象化した構造を表現している。図 5.6 データ保護システムモデルのクラウド実装例のクラウドゲートウェイが図 4.10 のゲートウェイサーバそのものに該当し，リモートサイトはクラウド，ターゲットボリュームはクラウドストレージで再現できる。

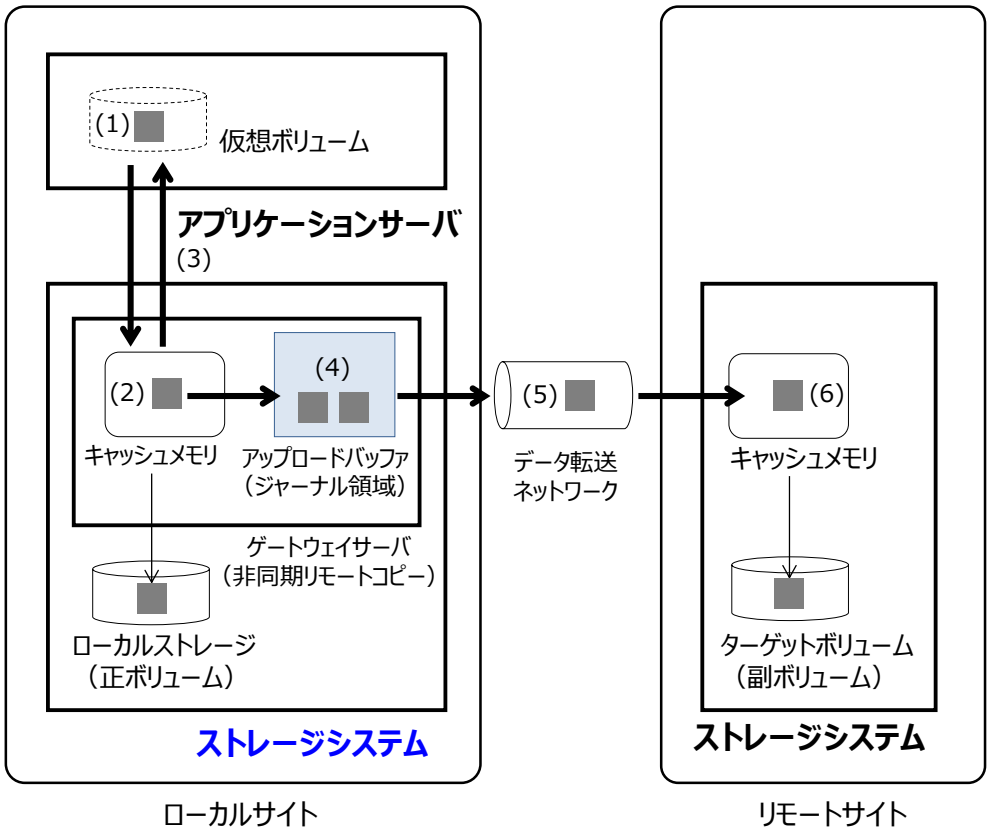



**図 5.5 データ保護システムモデルのストレージ実装例**

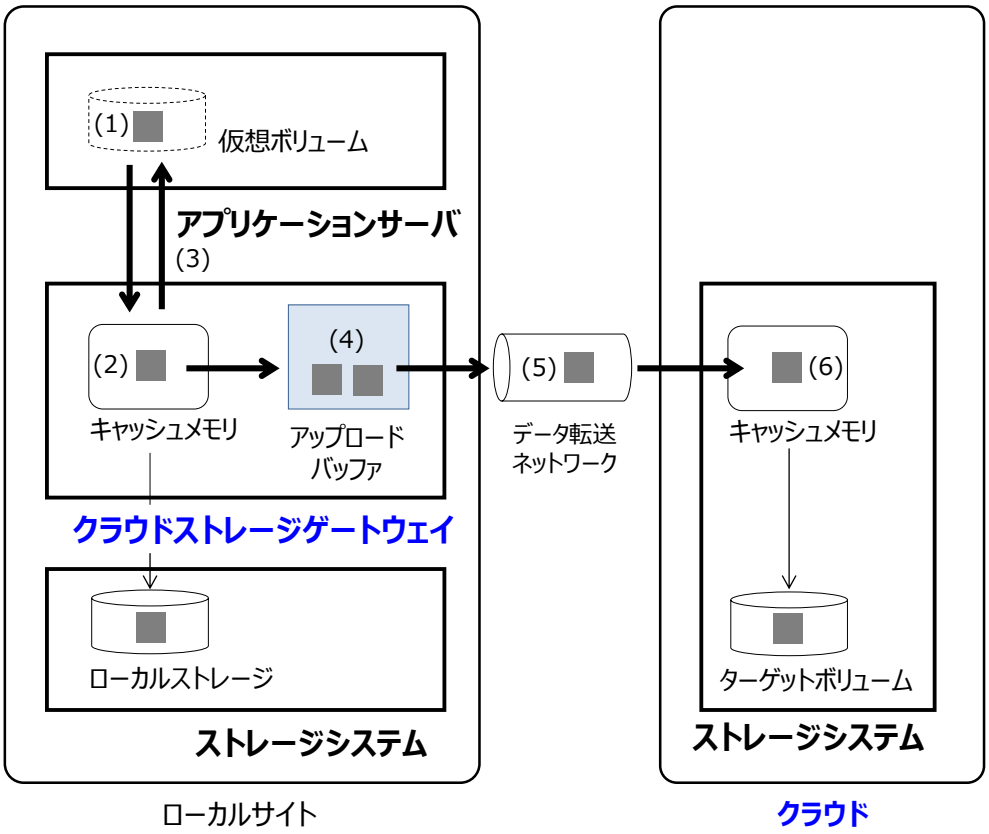



**図 5.6 データ保護システムモデルのクラウド実装例**

### 5.3.2. リカバリポイント計算方式

図 5.1 に示したとおり，アップロードバッファに滞留する未転送データは，インシデント発生時に消失する状態にある。これは言い換えると，未転送データのうち最も古いデータが書き込まれた時刻の，直前時刻までに生成されたデータはリモートサイトに転送済みであり，保護された状態であることと同義である。したがって，あらゆる時刻におけるリカバリポイントは，アップロードバッファに蓄積された未転送データのうち，最も古いデータの書き込み時刻に近似すると見なせる。そこで本計算では，この最も古いデータを発見する手順を定義する。図 5.7 に書き込みデータ量（変数 *In*）および未転送データ量（変数 *C*）の時系列推移の様子を例示する。書き込みデータ量が大きく，データ転送性能が追いつかなくなるとアップロードバッファに未転送データが蓄積されることを表現している。

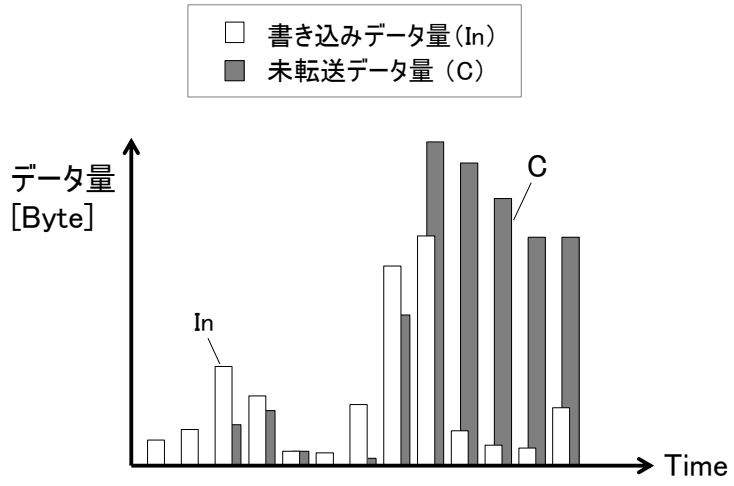



図 5.7 書き込みデータ量と未転送データ量の時系列推移例イメージ

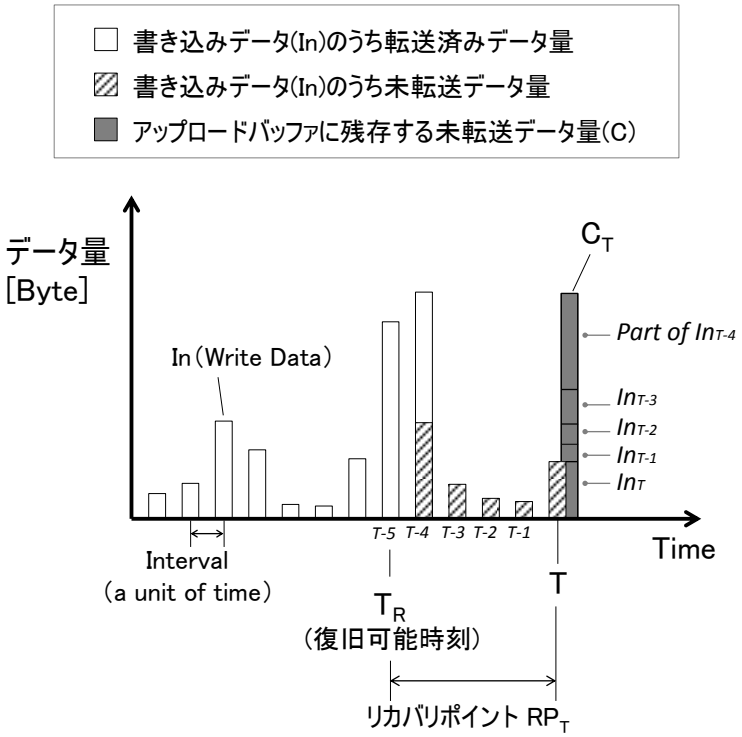



図 5.8 リカバリポイント

さらに図 5.8 は，ある時刻 *T* における未転送データ量 $C_T$ と，書き込みデータ量 $In_T$ の関係を表現している。時刻 *T* において最も古い未転送データの書き込み時刻は，時刻 *T* からさかのぼって書き込みデータ量 *In* を累加した値が，未転送データ量 $C_T$ に達した時刻に一致する。すなわち図 5.8 における斜線部で表現した累加値は，時刻 *T-4* の時点で $C_T$ に達する。この時刻 *T* でインシデントが発生すると，時刻 *T-4* から時刻 *T* までにサーバから書き込まれたデータは未転送であるため消失する。しかしその直前の時刻である *T-5* に書かれたデータはリモートサイトに記録されているため，復旧可能である。そのため復旧可能時刻は時刻 *T-5* となる。以上の考察により，時刻 *T* におけるリカバリポイント $RP_T$ は *T* と *T-4* の差に一致する。これらの手続きは以下の数式で表現できる。

$$RP_T = \begin{cases} 0, & C_T = 0 \\ (n+1) \times Interval, & C_T > 0 \end{cases}$$

ただし，*n* は以下を満たす最小の整数である。

$$C_T \leq \sum_{i=0}^{n} In_{T-i}$$

*Interval* は観測したデータのサンプリング間隔を表す。3.4.3 で述べたとおり，*In* および *C* を含むシステム監視ログは定数 *Interval* を単位とする時系列のデジタルデータである。*n* は時刻 *T* からさかのぼって *In* を累加した回数に相当する。

### 5.3.3. 性能評価の方法

調査対象の時間帯を通して，計算したリカバリポイント $RP_T$ と RPO を比較し，検査対象のシステムが性能目標を達成していたかどうかを判定する。$RP_T$ が RPO より短時間であれば，同システムの性能要件は達成されていたと見なせる。逆に $PR_T$ が RPO より長時間であれば，性能要件が未達成であり，性能を増強してデータ消失のリスクを低減するための対策が必要となる。

# 6章　データ保護システムのサイジング技術の提案

## 6.1. データ保護システムサイジングにおける課題

前節の性能評価を適用した結果，リカバリポイントの最大値が RPO より長時間であった場合は，目標を達成するようにシステム構成を見直すことが望ましい。逆にリカバリポイントが RPO と比較して著しく短い時間であった場合，システムリソースを減設することで同システムにかかるコストの削減を期待できる。

性能目標の達成とコスト最小化の二つの要件を実現するために，データ転送ネットワークの帯域をはじめとするシステムリソースを最適なサイズに調整することが求められる。このシステムリソースのサイズの調整にあたり，リカバリポイントを予測する手段がないことが課題となる。そこで本研究では，データ保護システムの振る舞いをシミュレーションで再現することで，リカバリポイントを予測する技術を提案する。3.3 で述べたとおり，一般にシステム構成の変更には一時的なサービス停止を要するため，多くの準備作業や機会損失のリスクを伴う。そのため，シミュレーションには十分に高い精度が求められる。

## 6.2. 課題の解決方針

適正システムサイズを探索する手順を図 6.1 に示す。本研究では，システムサイズを直接求める計算式を定義するのではなく，いくつかの想定のうち最適なシステム構成を選択する帰納的アプローチをとる。(1)まずデータ保護システムのリソース量をチューニングしたいくつかの想定構成を設計する。次に，(2)各構成において，同一の書き込みデータ量が発生した場合のリカバリポイントをシミュレーションにより算出する。(3)その結果 RPO を達成し，かつリソース量に依存するシステムコストの和が最小となる構成を最適システムサイズとして導出する。

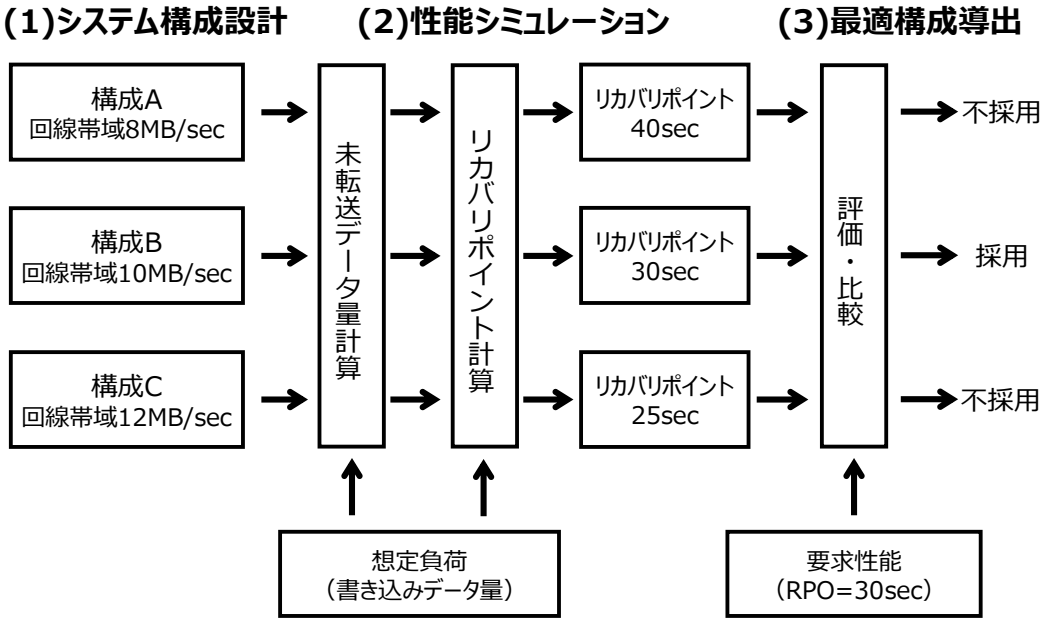



**図 6.1 システムサイジングのアプローチ**

リカバリポイントのシミュレーションにあたっては，アップロードバッファに滞留する未転送データ量を計算する。図 6.2 に示すとおり，書き込みデータ量の実績と，予測した未転送データ量をリカバリポイント計算式(i)の入力とすることで，リカバリポイントを算出することが可能となる。

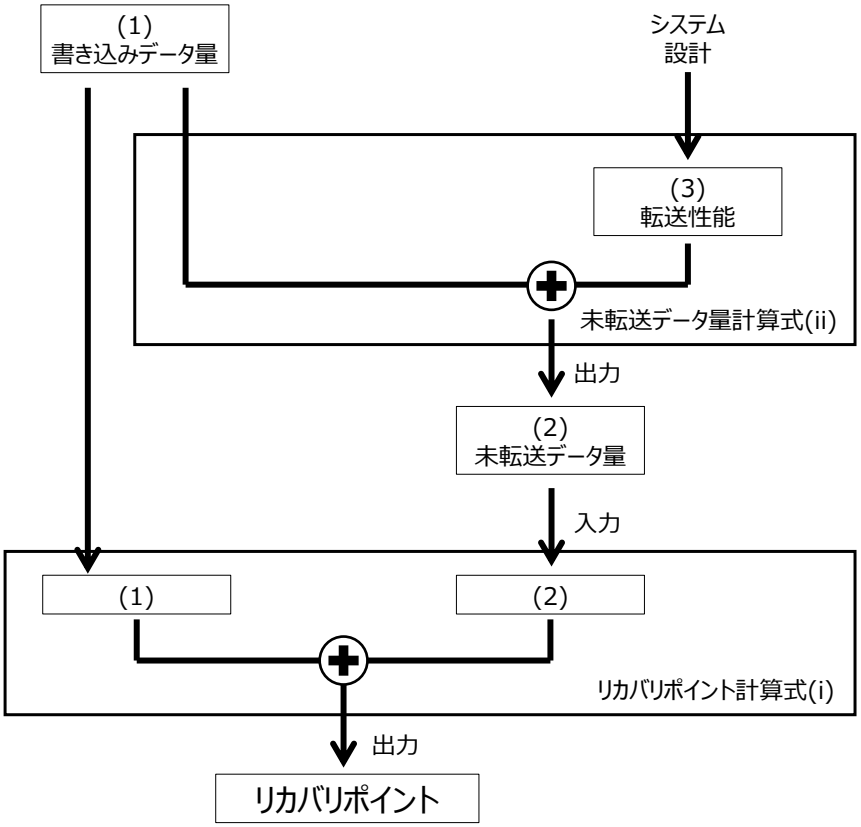



図 6.2 性能シミュレーションの入出力フロー

## 6.3. データ保護システムサイジング方式

### 6.3.1. システム構成のサンプリング

システム構成設計の段階では, 変更できる様々なリソースパラメータを考察するべきである。ネットワーク帯域だけでなく，ゲートウェイサーバの CPU コア数やアップロードバッファのストレージ容量などの調整により, データ保護システムの性能をチューニングできる。

本研究では特に, データ保護システムの性能を支配するネットワーク帯域を変数としたシステム構成チューニングについて検証する。

### 6.3.2. 未転送データ量計算方式

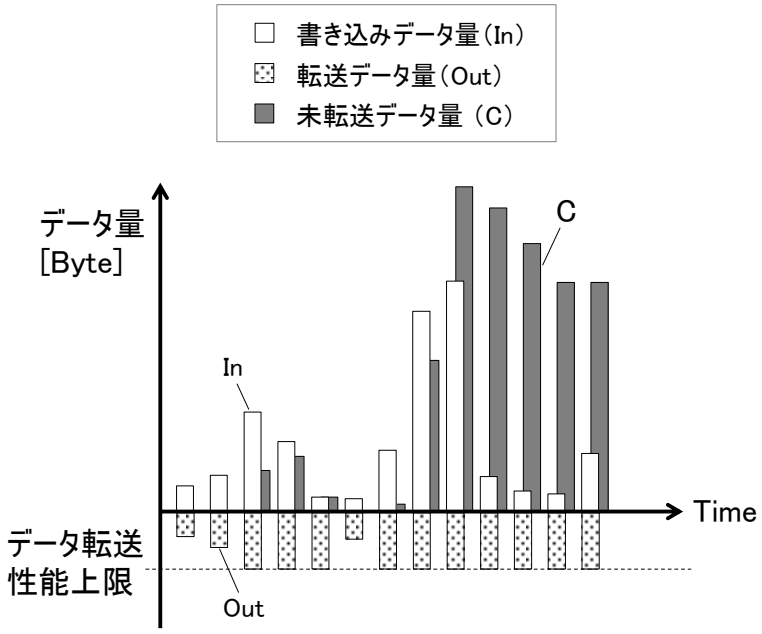



**図 6.3 未転送データ量計算方式**

図 6.3 は，書き込みデータ量 *In*，転送データ量 *Out* および未転送データ量 *C* の時系列推移の例を表現している。これらの関係は以下の式で表すことができる。

$$C_T = C_{T-1} + In_T - Out_{T-1}$$

ここでは時刻 *T* における書き込みデータ量を $In_T$，ゲートウェイからリモートサイトへ送出されたデータ量を $Out_T$ と表記している。同式はすなわち，書き込みデータ量 $In_T$ に対して転送データ量 $Out_T$ が追いつかなくなると，未転送データ量 $C_T$ が増加することを表している。$C_{T-1}$ は時刻 *T* より一つ前の時刻 *T-1* における未転送データ量を表す。

転送データ量 $Out_T$ は時刻 *T* における転送対象データ量に対して転送性能が不足する場合，同性能が上限となる。図 5.4 のモデルにおいて，ゲートウェイサーバがデータを

できるだけ滞留させず，ただちに転送する前提においては，時刻 *T* における書き込みデータ量 $In_T$ と，その直前の時刻 *T-1* にアップロードバッファに滞留していたデータ $C_{T-1}$ が同時刻 *T* における転送対象データ量となる。

また，ゲートウェイがバッファに入ってきたデータを直ちに送出するのではなく，書き込まれたデータをアップロードバッファに一定時間保管した後，まとめてリモートサイトへ配信する振る舞いも想定される。この一時保管時間を定数 *D* と表すと，時刻 *T* における転送対象データ量は，時刻 *T-D* における未転送データ量 $C_{T-D}$ が相当する。

バッファの一時保管に関する振る舞いの違いを表 6.1 に示す。

**表 6.1 バッファデータの一時保管とデータ転送処理の振る舞いの違い**

| # | 実装 | 一時保管時間 *D* | リカバリポイント | データ圧縮効果 | 時刻 *T* の転送対象データ |
|---|---|---|---|---|---|
| 1 | データベース，ストレージ | マイクロ秒，ミリ秒 | 優る（短い） | 劣る（小さい） | $In_T + C_{T-1}$ |
| 2 | サーバ仮想化ソフト，クラウドゲートウェイ | 分オーダ | 劣る（長い） | 優る（大きい） | $C_{T-D}$ |

加えて転送性能はアップロードバッファの入出力性能 $P_{Buffer\_IO}$ あるいはデータ転送ネットワーク性能 $P_{Network}$ のいずれか低い方が上限となる。以上の条件により，時刻 *T* における転送データ量 $Out_T$ は以下のように定式化される。

$$Out_T = \begin{cases} \min\{In_T + C_{T-1}, P_{Buffer\_IO}, P_{Network}\}, & D = 0 \\ \min\{C_{T-D}, P_{Buffer\_IO}, P_{Network}\}, & D > 0 \end{cases}$$

転送データ量 $Out_T$ の計算にあたっては，データ圧縮もしくは重複排除の効果を考察に加えることで，より精度を高められる可能性がある。アップロードバッファに滞留す

るデータのうち，同一アドレスへの上書きを検出できれば，すべてのデータを送信するのではなく，最新データのみを送ることで転送効率を上げることができる。あるいはゼロが連続するホワイトスペースの削除など，圧縮のアルゴリズムはいずれの手法でも構わない。時刻 $T$ における転送対象データの圧縮率を $Z_T$ と表記した場合，転送できる圧縮前データ量は性能限界を $Z_T$ で除算した値に一致する。

$$Out_T = \begin{cases} \min\left\{In_T + C_{T-1}, \frac{P_{Buffer\_IO}}{Z_T}, \frac{P_{Network}}{Z_T}\right\}, & D = 0 \\ \min\left\{C_{T-D}, \frac{P_{Buffer\_IO}}{Z_T}, \frac{P_{Network}}{Z_T}\right\}, & D > 0 \end{cases}$$

また一般的なインターネットでは様々なオーバヘッドが発生するため，通信性能が設計値よりもある程度低下することが想定される。ベストエフォート型のネットワークでは，他者のトラフィックの影響によって通信性能が低下する可能性がある。例えば 10Mbps，1Gbps といったカタログスペックであっても，それよりも劣化した値が実際の通信性能の上限となる。そこでデータ転送性能 $P_{Network}$ は，通信効率 *Transmission_Efficiency* パラメータを考慮して決定する。

$$P_{Network} = Bandwidth \times Transmission\_Efficiency$$

### 6.3.3. 適正システムサイズの導出

6.3.2 節の手続きにより，未転送データ量をシミュレーションによって予測することが可能となった。さらに図 6.2 に述べたとおり，これを数式(i)の入力とすることでリカバリポイントを予測できる。

図 6.4 は，データ転送ネットワークの帯域を調整した 6 通りのシステム構成について，リカバリポイントを算出した例である。この例は，転送性能の増加に比例してリカバリポイントが短くなる様子を表している。これらの構成のうち，リカバリポイントが RPO より短時間になるケースが採用候補となる。一方で過剰な性能はコスト増の要因となるため，6 通りの想定のうち，リカバリポイントが RPO を達成し，かつコストが最小と

なる構成を採用することが望ましい。図 6.4 のとおり RPO が 35 秒前後に定義されていれば，リカバリポイントがこれを達成し，ネットワークコストが最小となる帯域 10Mbps となる構成を適正サイズとして採用するべきである。

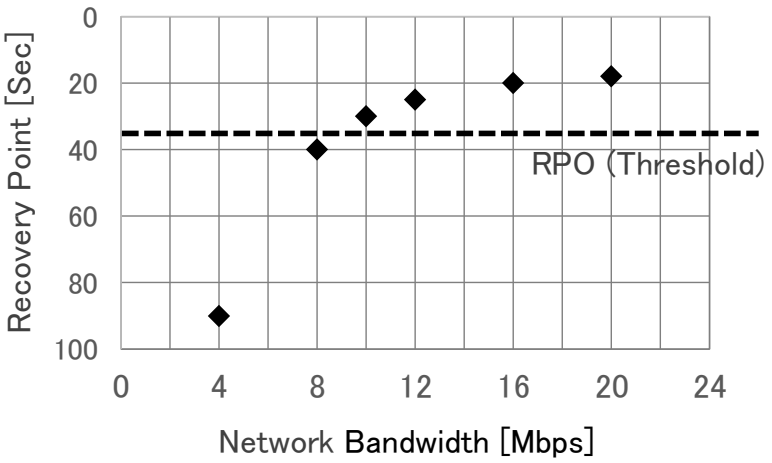



**図 6.4 適正システムサイズの導出**

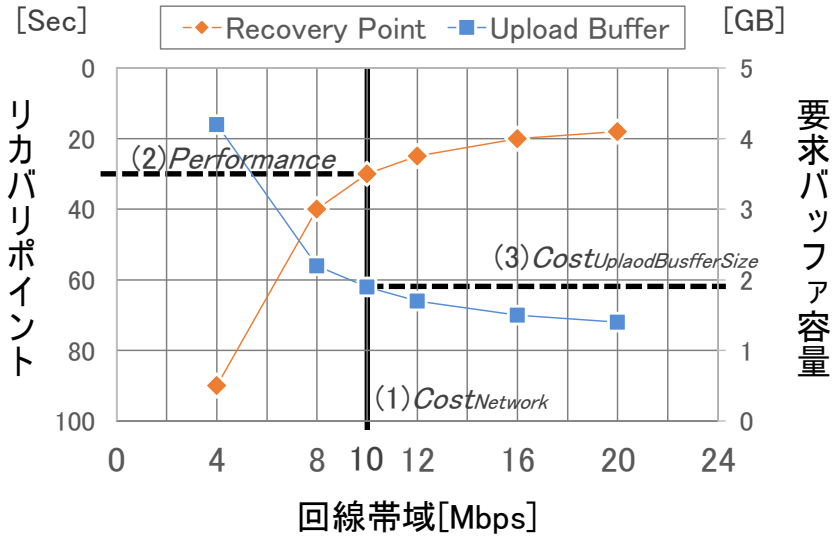



**図 6.5 システム性能とコストの関係の例**

また，本研究のシステムモデルによれば，データ転送ネットワークの帯域が大きいほ

どバッファに蓄積する未転送データ量が減少し，帯域が小さいほど未転送データ量が大きくなる。そのため，システムコストの決定要因には，未転送データ量を保管するアップロードバッファの容量を含めるべきである。シミュレーションによって同時に出力される，リカバリポイントと未転送データ量の関係を図 6.5 に例示する。

データ転送ネットワークの帯域とリカバリポイントの短さ（すなわち性能の良さ）は比例関係である。これに対し，ネットワーク帯域と，未転送データの格納に必要とされるアップロードバッファの容量は反比例の関係となる。一例として，図 6.5 の回線帯域が 10Mbps の構成では，リカバリポイント最大値が 30 秒，未転送データ量の最大値が約 2GB であることを示している。このケースでは，少なくとも 2GB のストレージをアップロードバッファに設ける必要がある。したがって，シミュレーションによって出力されるアップロードバッファにかかるコスト $Cost_{UploadBufferSize}$ を変数に加える。

システム全体のコストは固定費と変動費の和で表すことができる [78]。このうち変動費はシステムリソースの変動によって発生するコストの合計値とする。図 6.5 の考察に倣い，変動コストは以下の数式で定義する。

$$Cost_{Variable} = Cost_{Network} + Cost_{UploadBufferSize}$$

以上をまとめると，本研究では，シミュレーションした構成のうち，以下の条件を満たす構成を適正システムサイズとして導出することになる。

- リカバリポイントが RPO を達成する構成
- 回線帯域コストとアップロードバッファコストの和 $Cost_{Variable}$ が最小となる構成

# 7章　提案手法の評価

## 7.1. ストレージリモートコピーシステムへの適用

### 7.1.1. システムサイジングの試行

本節では，ストレージの非同期リモートコピーシステムを想定し，6 章で述べたシステムサイジング方式の適用を試行する。試行を通じて同方式による未転送データ量計算式ならびにリカバリポイント計算式の適用可否を調査する。

図 4.9 に述べたとおり，ストレージのリモートコピーは特に基幹系システムの重要データを保護するための機能であり，書き込まれたデータをできるだけ早くリモートサイトに送出しようとする。よってキャッシュからアップロードバッファへの出力をはじめとする内部処理はミリ秒オーダ以下と見込める。一方で図 6.3 に述べたとおり，本研究のシミュレーションは運用管理データのサンプリング間隔を単位としてカウントするため，一般的には秒あるいは分オーダとなる。ストレージ内部で発生するミリ秒オーダの処理遅延は，その時間軸と比べて十分に短く，誤差の範囲であるため無視して問題ない。したがって一時保管時間 $D$ はゼロに近似する値と解釈し，未転送データ量計算(ii)に適用する転送データ量 $Out_T$ には以下の計算式を適用する。

$$Out_T = \min\{In_T + C_{T-1}, P_{Buffer\_IO}, P_{Network}\}$$

リカバリポイント計算の入力とした書き込みデータ量の時系列推移を図 7.1 に示す。引用元である Cello 1999 は，実際に企業のデータセンタで発生したストレージシステムへのリード･ライト実績の一般公開情報であり，本計算の試行には十分なデータである。ここでは，あるボリュームにおける書き込みが局所的に増加した事象に着目し，その前後を含めた時間帯を対象とした。具体的には，15 秒あたり最大 300MB を超える書き込みが発生した 0 時 00 分から 8 時 00 分の区間を切り出して試行対象とした。このような

事象はアップロードバッファの未転送データ量が増加する要因であり，方式検証に有効である。

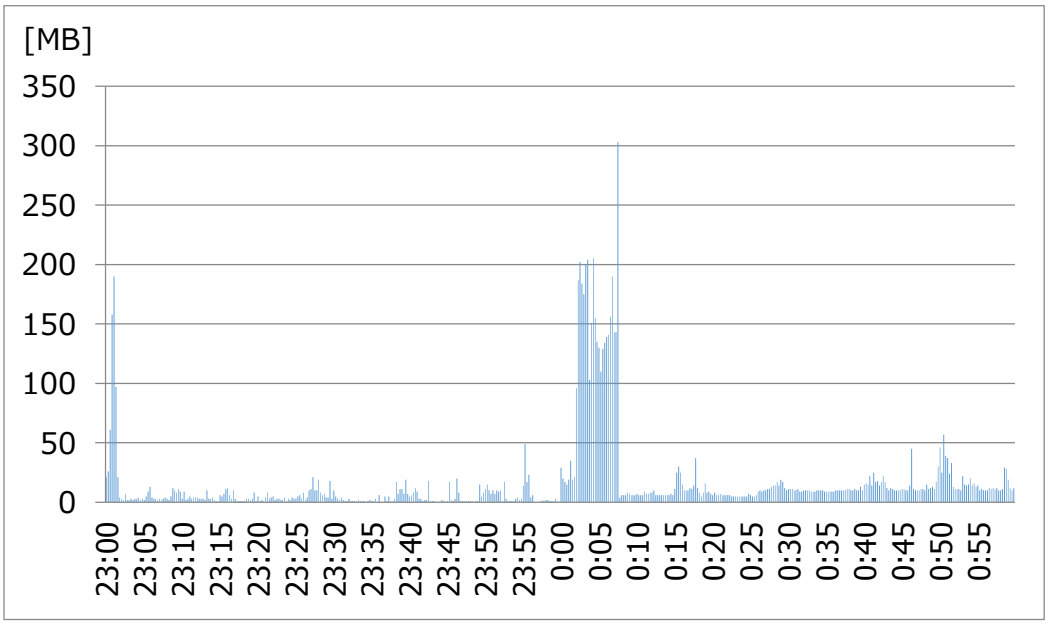



**図 7.1 書き込みデータ量**

次に，データ転送ネットワークの帯域を３通りに設定した条件で，未転送データ量計算式(ii)を適用した計算結果を図 7.2 に示す。ネットワーク帯域を①6MB/sec, ②8MB/sec, ③10MB/sec と設定したところ，未転送データ量の最大値はそれぞれ①1645MB，②979MB，③319MB と算出された。さらに，図 7.3，図 7.4，図 7.5 に示すとおり，各ケースにおけるリカバリポイントはそれぞれ①285sec，②135sec，③45sec と算出された。なお，本計算において転送効率変数 *Transmission_Efficiency* は 100%の条件で試算した。

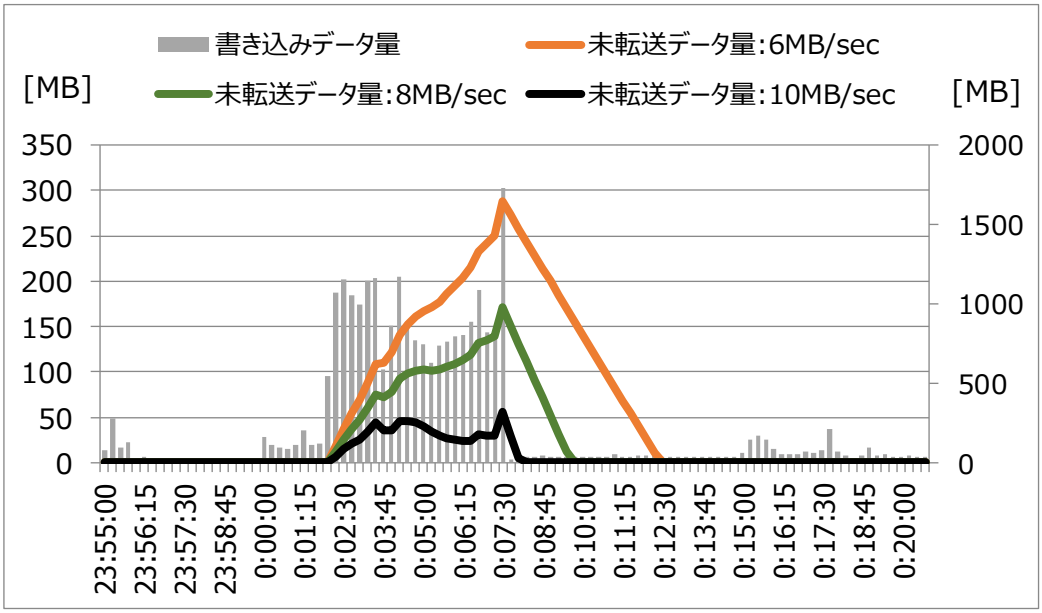



図 7.2 未転送データ量の計算結果

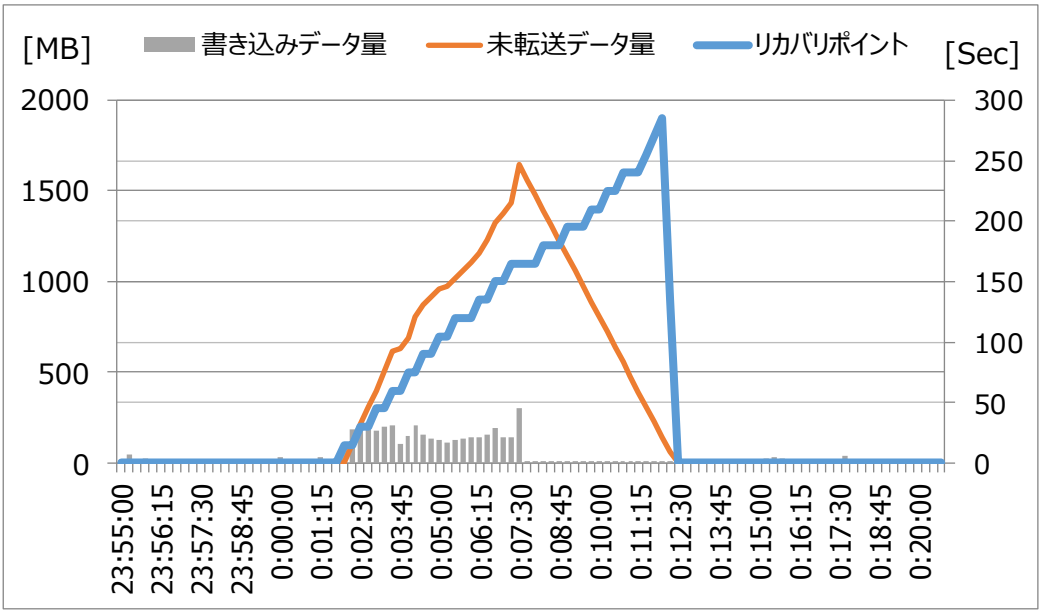



図 7.3 未転送データ量とリカバリポイント（回線帯域 6MB/sec）

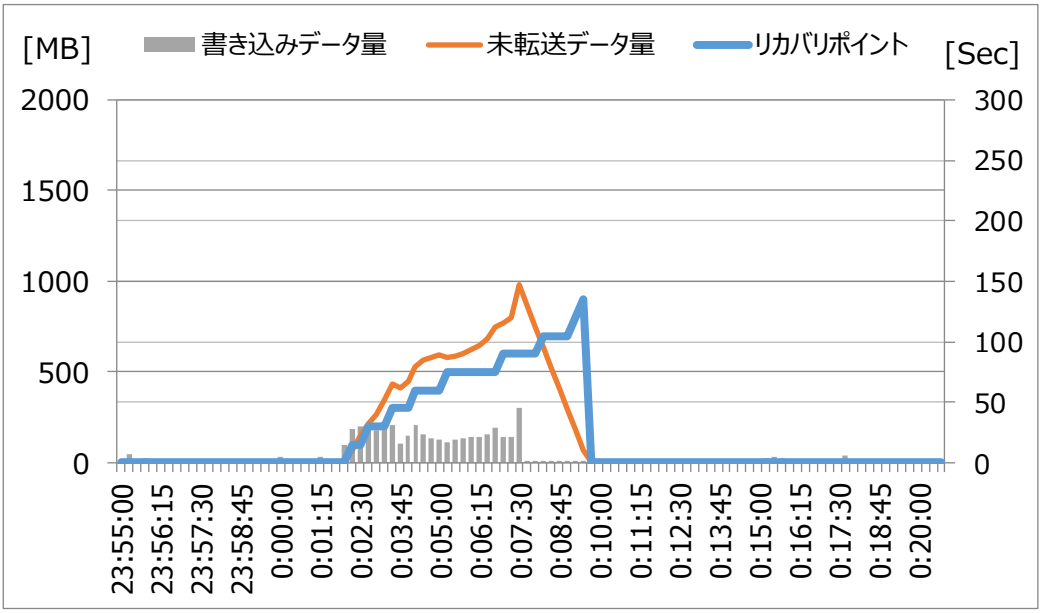



**図 7.4 未転送データ量とリカバリポイント（回線帯域 8MB/sec）**

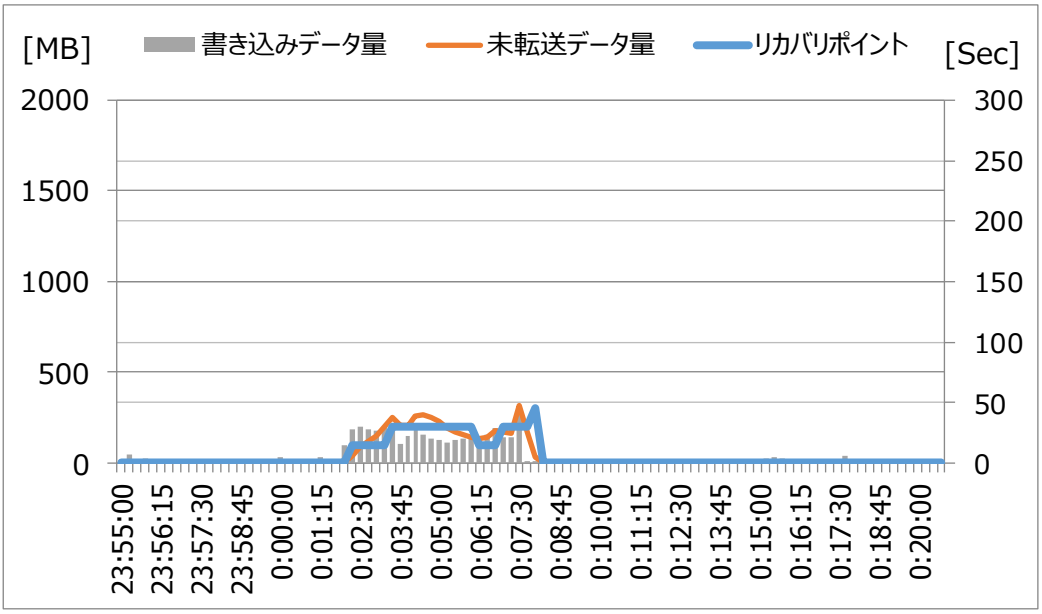



**図 7.5 未転送データ量とリカバリポイント（回線帯域 10MB/sec）**

前述した３通りの構成に加え，回線帯域 16MB/sec および 24MB/sec の構成を追加した未転送データ量とリカバリポイントの試算結果を図 7.6 に示す。同一のデータ書き込

み負荷が発生する条件において，最大リカバリポイントは帯域 6MB/sec 構成における 285sec から，16MB/sec 構成の 15sec と徐々に改善し，24MB/sec の構成ではゼロに達した。逆に反比例する形で未転送データ量は少なくなり，最大 1645MB から最小値ゼロへと推移した。なお，4.4.4 節に前述のとおり非同期リモートコピーではサーバへ応答した後でリモートサイトへデータ転送する仕組みであるため，厳密には「同期コピー」と同義である「リカバリポイント＝ゼロ」という状態にはなり得ない。しかしながらアップロードバッファでの滞留が起こらない状況では，データ転送遅延をネットワークの通信処理時間のみに抑えられる。これは運用管理データの観測単位であるサンプリング間隔（秒～分オーダ）と比べてはるかに短時間であるため，誤差の範囲として取り扱う方針とする。

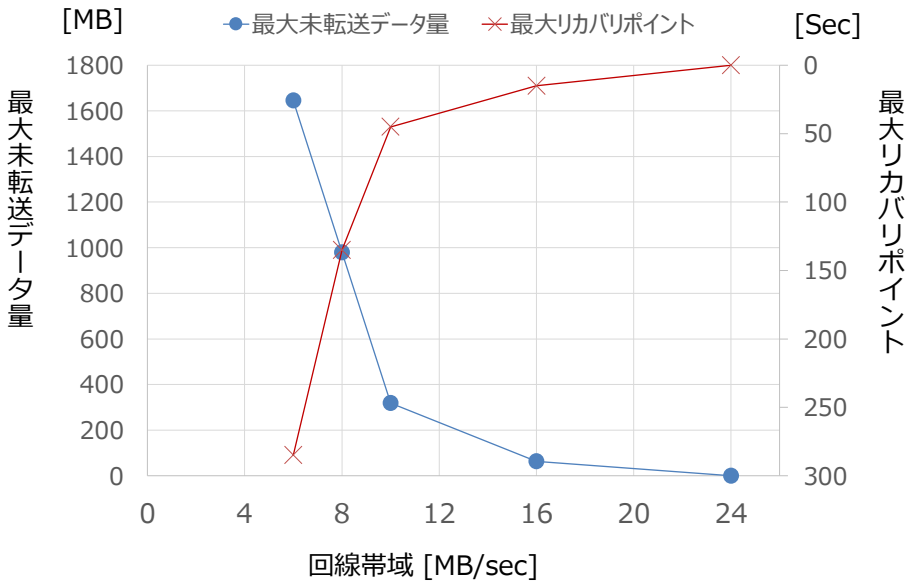



**図 7.6 非同期リモートコピーシステムのデータ保護性能シミュレーション**

これらのシミュレーション結果をコスト換算した結果を図 7.7 に示す。ここではデータ転送ネットワーク回線帯域を 100Mbps/月あたり 320,000 円，アップロードバッファのストレージを本稿執筆時点の SSD（Solid State Disk）価格から 1GB あたり 72 円と仮定

した。換算したシステムコスト $Cost_{Variable}$ は，更改後 6 ヶ月分の回線帯域コスト $Cost_{Bandwidth}$ と最大未転送データ量を格納できるアップロードバッファ容量 $Cost_{UploadBufferSize}$ の和で算出した。本試算においては，未転送データの蓄積量がゼロ MB から 1.7GB 弱と小さいため，ストレージコストに対して回線コストが支配的となり，ほぼ回線帯域に比例してコストが増加する結果となった。したがって，リカバリポイントが RPO を達成する構成のうち，最も帯域の小さい構成を適正システムサイズとして選択すれば良い。

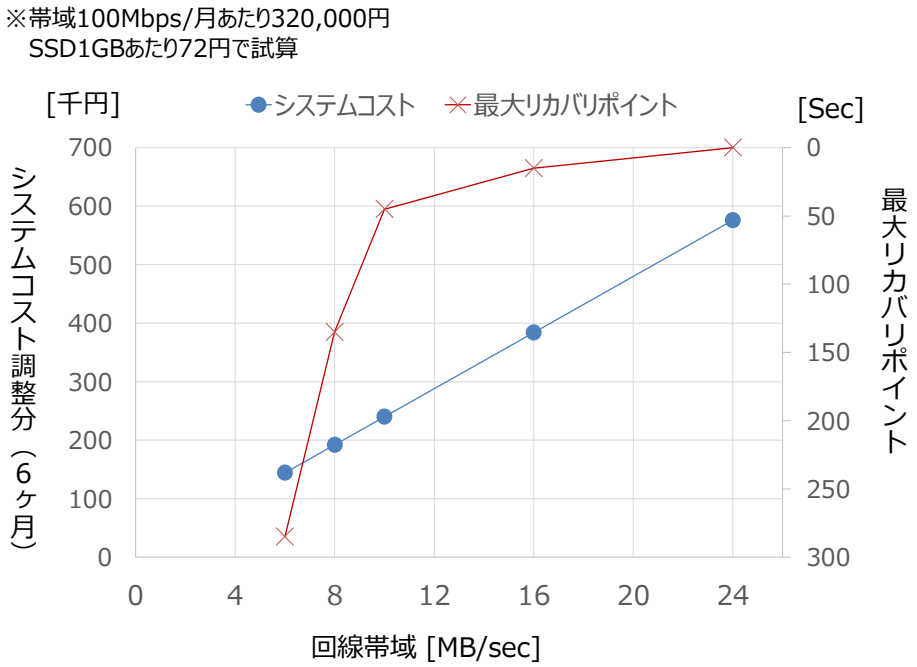



**図 7.7 非同期リモートコピーシステムのコスト試算結果**

一例として，図 7.8 に例示するように RPO が 120 秒であった場合，それよりもリカバリポイントが短時間となる構成（帯域 10MB/sec，16MB/sec，24MB/sec）のうち，システムコストが最小となる帯域 10MB/sec の構成を適正サイズとして出力する。

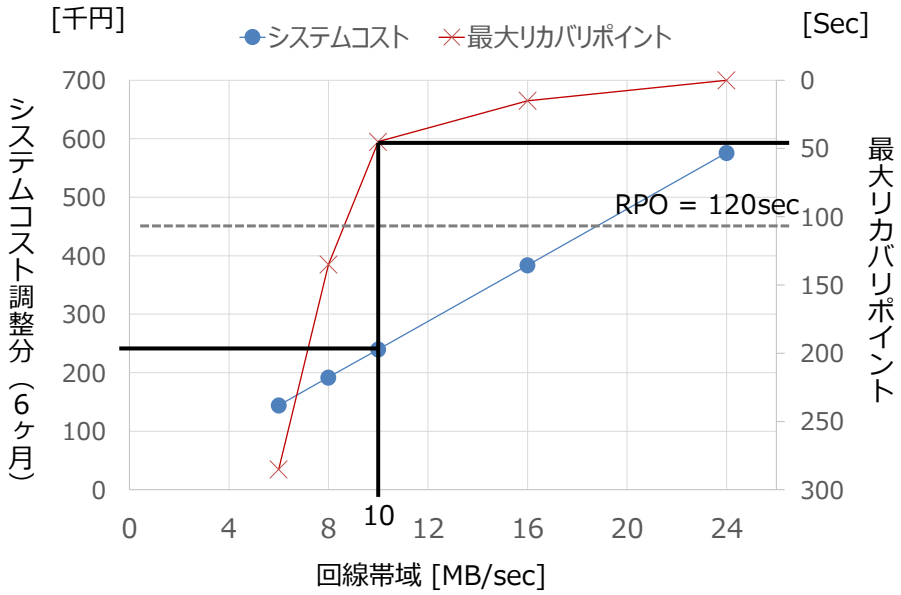



**図 7.8 適正システムサイズ選択の例 （RPO 120sec の場合）**

## 7.2. クラウドゲートウェイシステムへの適用

### 7.2.1. クラウドゲートウェイシステム実証実験

図 5.6 に述べたとおり，本研究の性能評価およびサイジング方式は，クラウドゲートウェイで実装したデータ保護システムに適用可能である。ここでは AWS Cloud Gateway を用いた実験を通じて，その有効性を検証する。

(1) 実験条件

本実験で構成したシステム構成を表 7.1 に示す。本実験では，企業データセンタに AWS Cloud Gateway をキャッシュ型ボリューム方式で実装した。データセンタから AWS へはインターネットで接続した。また同データセンタは関東圏で稼働しているため，AWS はネットワーク遅延の影響が少ない東京リージョンを選択した。AWS からはターゲットボリュームとなる記憶領域をストレージサービスである S3 で供給した。以上の構成で Cloud Gateway は AWS S3 に接続し，これをさらに仮想デバイスにマッピングし

た。

仮想デバイスは iSCSI プロトコルでオンプレミス環境のサーバに公開し，サーバのファイルシステムで仮想ボリュームをマウントし，リード･ライト可能な記憶領域とした。サーバもまた同データセンタで稼働する VMware の仮想マシンで構成した。サーバの OS には Cent OS を採用した。

Cloud Gateway は VMware の仮想インスタンスで実装した。アップロードバッファにはハードディスクを採用し，また AWS へ接続するネットワークの性能は 10Mbps にパラメータ設定した。

**表 7.1 実験用クラウドゲートウェイの構成**

| | |
|---|---|
| vSphere Version | 7 |
| OS | Cent OS |
| CPU | 4 仮想 CPU （323MHz） |
| メモリ | 8GB |
| ローカルストレージ | 80GB （SSD） |
| アップロードバッファ | 32GB （SSD） |
| AWS への転送性能上限 | 10Mbps |

実験にあたっては，実際に企業データセンタで発生した書き込み負荷を再現した。元となったワークロードは，トランザクションを処理するデータベースシステムである。この負荷を再現するために，仮想デバイスをターゲットとした書き込みを発生させた。書き込みの発生にはオープンソースの I/O 性能ベンチマークツールである fio を使用した。負荷の再現においては，過去に発生した書き込みデータ量（Sequential Write MB/sec）と同量の書き込みを発生させた。この実験では書き込みデータ量だけ再現し，書き込みの内容は再現しない。そこで転送データの圧縮が発生しない条件（圧縮率ゼロパーセント）での実験を行った。

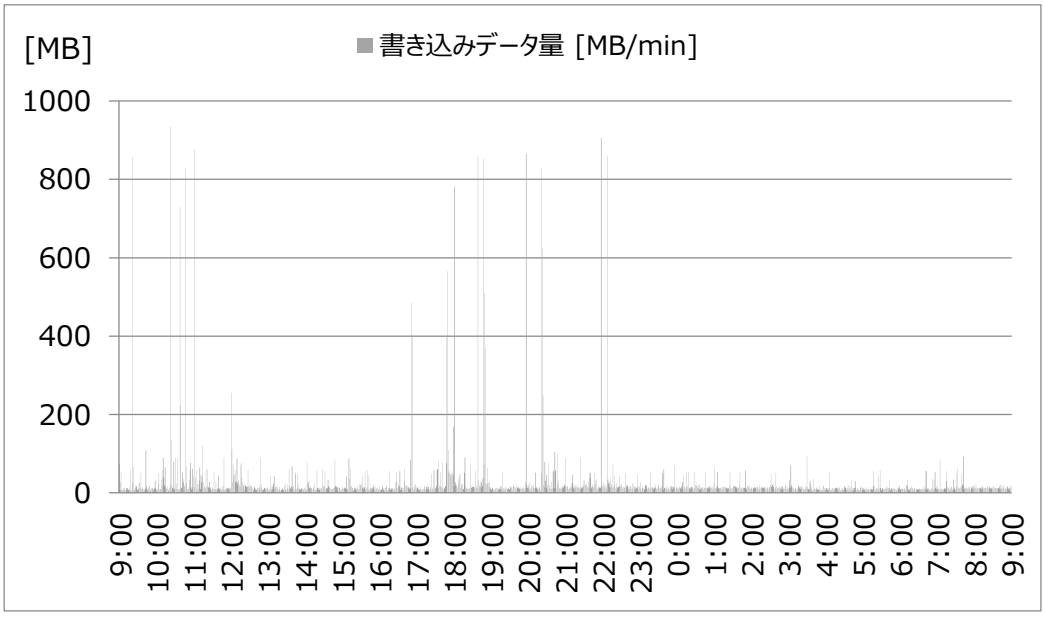



**図 7.9 書き込みデータ量**

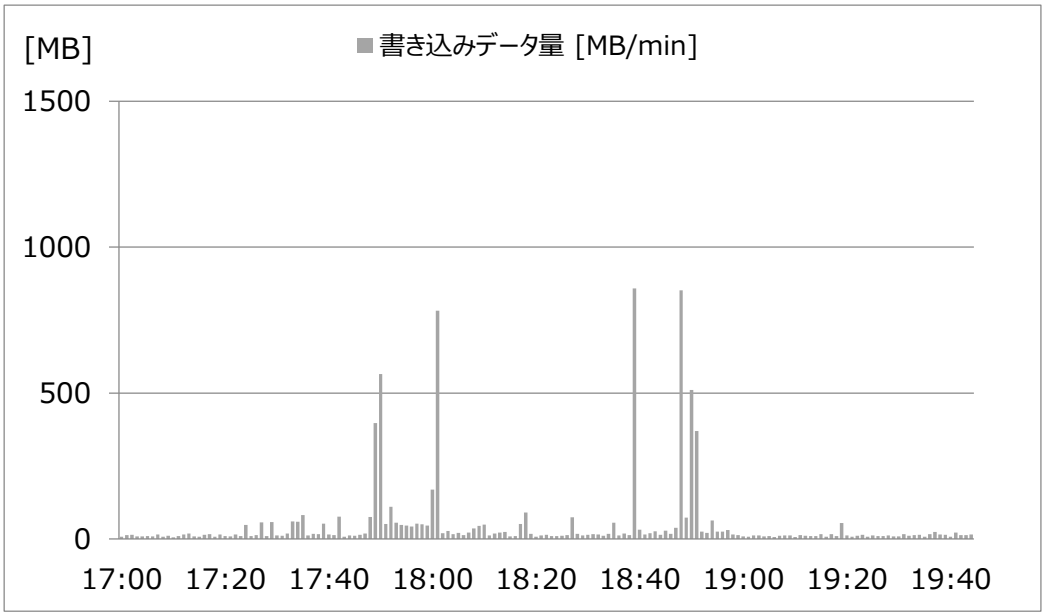



**図 7.10 書き込みデータ量（評価対象区間抜粋）**

再現した書き込みデータ量の時系列推移を図 7.9 および図 7.10 に示す。本実験では仮想ボリュームへの書き込みを 1 分ごとに変動させた。このうち特に負荷の変動が生じた,

17 時 00 分から 19 時 45 分までの区間を評価対象とした。また，図 7.10 の負荷を発生させたときの負荷を Cloudwatch で観測した結果を図 7.11 に示す。図 7.11 の観測値は，Cloudwatch のサンプリング間隔である５分の間に発生した書き込みデータ量の合計値（SUM）を表現している。

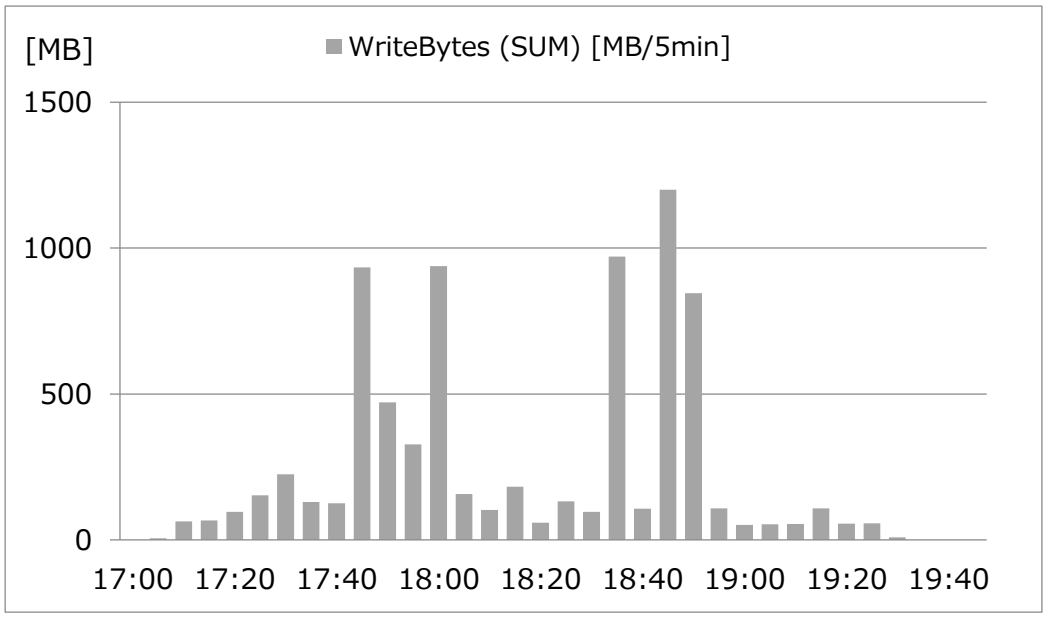



**図 7.11 書き込みデータ量の測定結果（サンプリング間隔５分）**

なお，AWS の監視サービスである Cloudwatch のメトリックと本研究の各パラメータの関係を表 7.2 に示しておく。

**表 7.2 Cloudwatch メトリックとの対応関係**

| # | 監視メトリックの意味 | 本研究の変数名 | AWS Cloudwatch のメトリック名 |
|---|---|---|---|
| 1 | 書き込みデータ量 | *In* | *WriteBytes* |
| 2 | 転送データ量 | *Out* | *CloudBytesUploaded* |
| 3 | 未転送データ量 | *C* | *QueuedWrites* |

(2) 実験結果

本実験による書き込みデータ量（*WriteBytes*）と転送データ量（*CloudBytesUplaoded*）の関係を図 7.12 に示す。

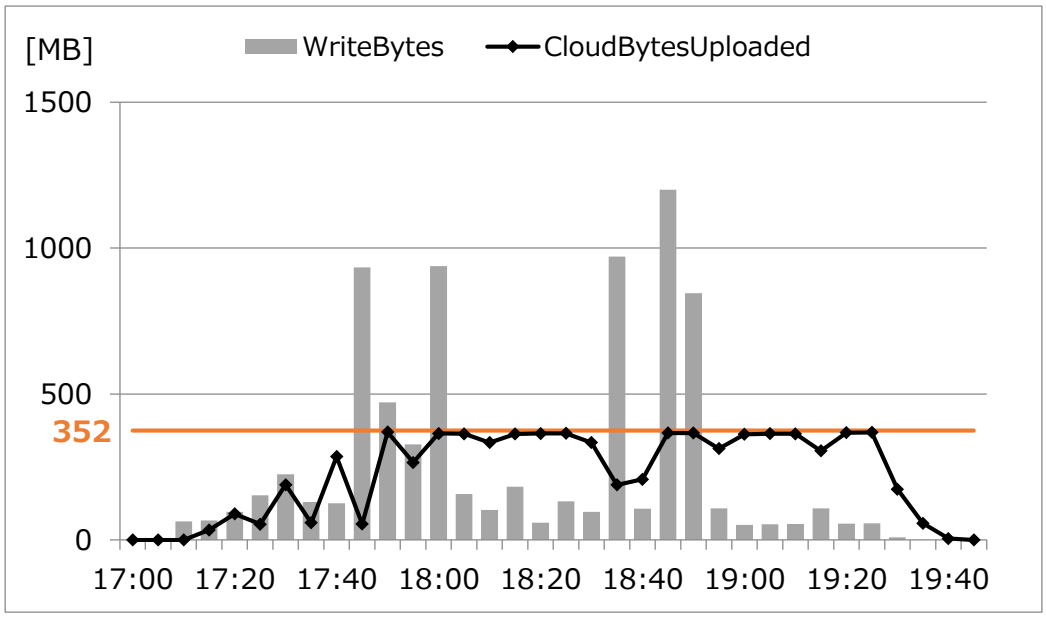



**図 7.12 実験結果（書き込みデータ量／転送データ量測定結果）**

本実験では，データ転送ネットワークの帯域を 10Mbps と設定した。10Mbps を単位換算すると，これは 375MB/5min と同値である。この条件において，転送データ量 *Out* に該当する *CloudBytesUploaded* は，５分あたり 352MB を上限として推移した。これは，同データ保護システムのデータ転送性能が 352MB/5min，すなわち 10Mbps の 94%であったことを表している。同結果より，転送効率 *Transmission_Efficiency* は 94%と推測される。したがって，ゲートウェイの通信帯域の設定値 10Mbps よりも，性能上限が６％低いことを前提として，以下の性能評価ならびにシミュレーションを試行する。

$$10\text{Mbps} \div 8\text{bit} \times 0.94 \times 60\text{sec} \times 5\text{min} = 352.5\text{MB}/5\text{min}$$

さらに，書き込みデータ量（*WriteBytes*）と転送データ量（*CloudBytesUploaded*），未転送データ量（*QueuedWrites*）の関係を図 7.13 に示す。転送性能上限 352MB/5min を上回るデータ書き込みが 17:45，18:00，18:35，18:45 とその近辺のポイントで発生している。性能が不足しアップロードバッファにデータが蓄積する様子が *QueuedWrites* の上昇から把握できる。この例では，18 時 50 分をピークとして 2.2GB を超える未転送データ量が観測された。この時点でオンプレミス環境に問題が発生すれば，2.2GB すべてのデータを消失することになる。アップロードバッファに滞留する未転送データは，転送性能がボトルネックとなるため時間をかけて排出される。18 時 50 分により後に書き込みデータ量は少量しか発生しないが，2.2GB 蓄積した未転送データ量 *QueuedWrites* は 5 分あたり最大でも 352MB しか減らないため，排出されるまでに数十分程度の時間を要している。この滞留時間こそが，データ転送遅延の原因であり，リカバリポイントが長時間となる要因である。

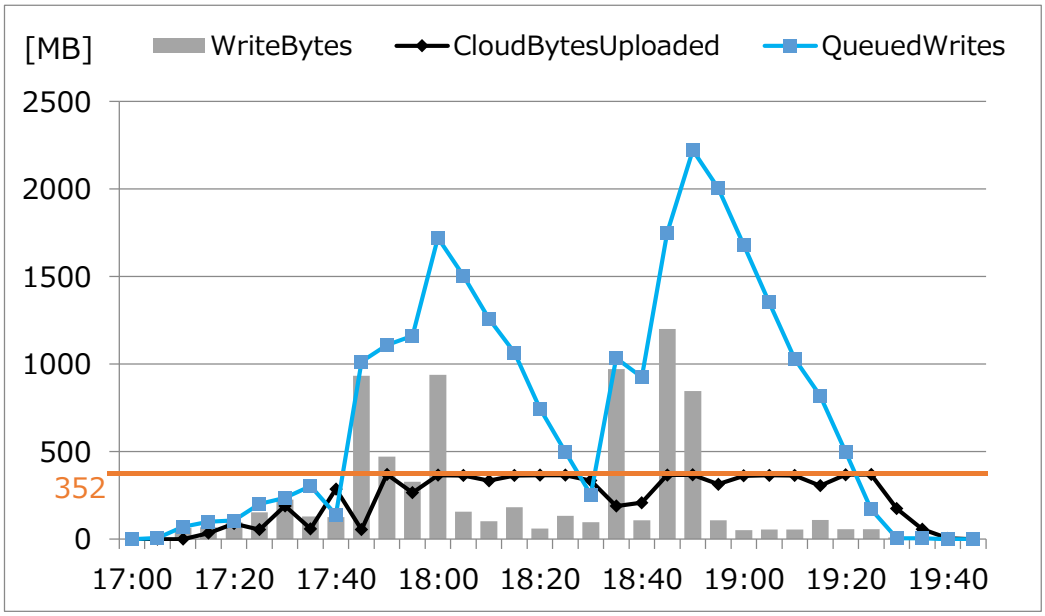



**図 7.13 実験結果（書き込みデータ量／転送データ量／未転送データ量）**

### 7.2.2. 性能評価方式の検証

図 7.13 に示した *QueuedWrites* の観測結果を数式(i)への入力として，算出したリカバ

リポイントを図 7.14 に示す。評価対象区間における最大のリカバリポイントは 19 時 20 分時点で「35 分」となった。これは 18 時 50 分に発生した書き込みデータがアップロードバッファに保存されてから 30 分後である 19 時 20 分までバッファに滞留していたことを表している。同システムの RPO が 35 分より長ければ性能要件を達成していたことになるが，もし RPO が 35 分より短い場合は，要件を達成するようにシステムサイズを変更しなければならない。あるいは RPO が 35 分より著しく長い場合も，同様に適正なシステムサイズに調整することで無駄なコストを抑制することができる可能性がある。

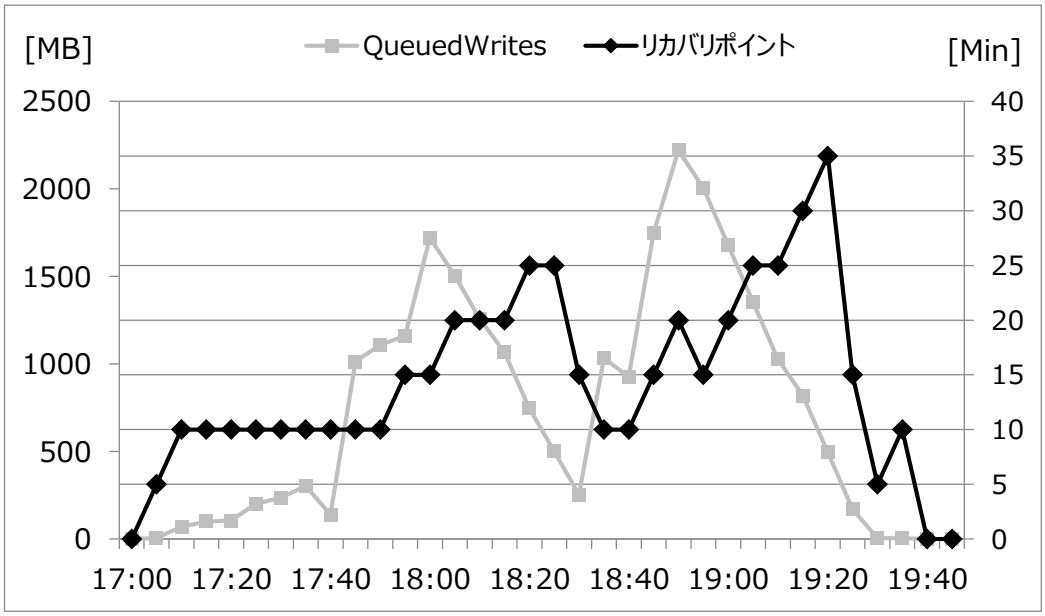



**図 7.14 リカバリポイント計算結果**

### 7.2.3. システムサイジング方式の検証

(1) 性能シミュレーション

適正なシステムサイズを発見するために, データ保護性能シミュレーションを実施する。本実験では，図 7.13 に示した実測値にシミュレーション結果をどれだけ近づけられるか検証する。なお転送効率 *Transmission_Efficiency* は，図 7.12 の実測をもとに推定

した値である 94%とした。また，アップロードバッファにおける一時待機時間 *D* は，AWS Cloud Gateway に組み込まれたパラメータであり，その仕様は開示されていない。加えて同パラメータは利用者には制御不可であるため推測に頼らざるを得ないが，図 7.12 の様子からこれを 5 分と仮定した。

以上の条件により，ネットワーク帯域を実験時と同じ 10Mbps としてシミュレーションを適用した結果を図 7.15 に示す。

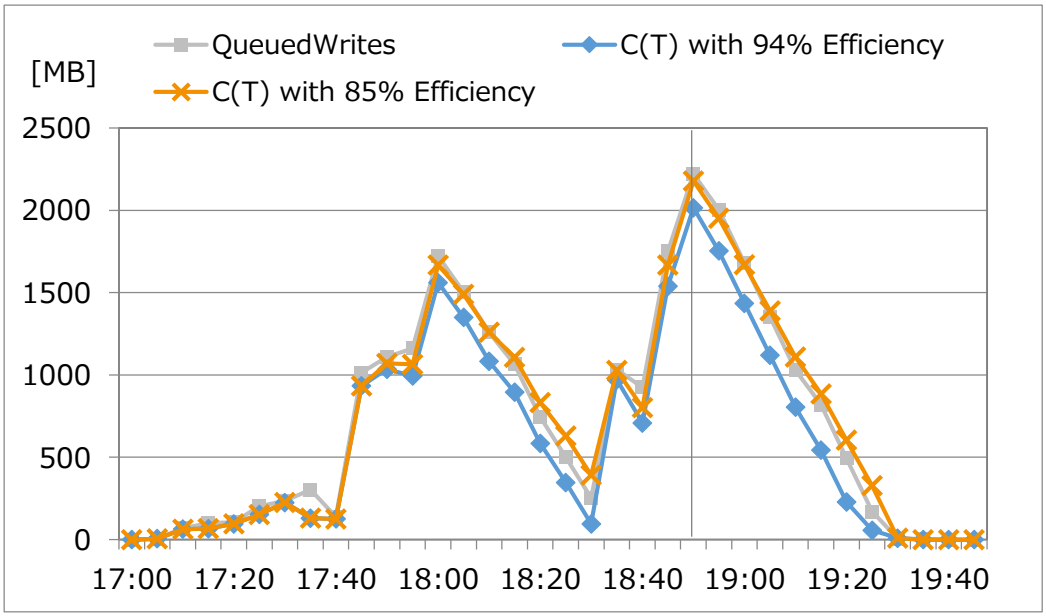



**図 7.15 未転送データ量計算結果**

本結果によれば，負荷がピークとなる 18 時 50 分における計算結果（2.01GB）と実測値（2.22GB）との誤差が 9.3%，検証対象データを通じた平均誤差が 17%となる結果が得られた。さらにシミュレーションの精度向上を図るため，通信効率 *Transmission_Efficiency* を 94%でなく 85%に調整したところ，同ピーク時における誤差を 1.9%，平均誤差をマイナス 4%まで抑えることができた。通信効率パラメータの調整による誤差の変動の様子を図 7.16 および図 7.17 に示す。本ケースの性能シミュレーションにおいては，誤差が最小となるよう通信効率を 85%に設定することが望ましいことがわかる。なお，本稿執筆時点では同パラメータを計算によって推測する手順の確立に

は至っておらず，図 7.18 図 7.17 に示す試行を繰り返して最適値を発見する手段を採用する。この通信効率パラメータの推定は今後の課題とする。以下，これらの考察から通信効率パラメータを 85%として検証を進める。


**図 7.16 通信効率と誤差の相関（ピーク負荷時点比較）**


**図 7.17 通信効率と誤差の相関（平均値比較）**

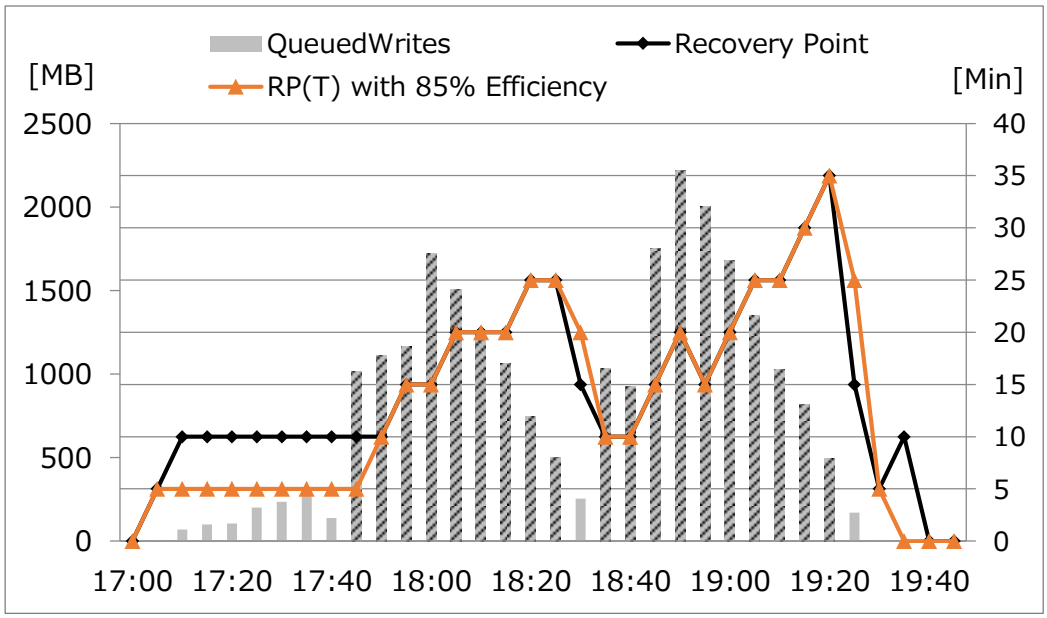



**図 7.18 リカバリポイント計算結果**

図 7.18 に示すとおり，リカバリポイントのシミュレーションについては，アップロードバッファに未転送データが蓄積されている 17 時 40 分から 19 時 20 分の間，実測値と計算値の誤差を 5 分以下に抑えることができた。5 分の誤差はシステム監視のサンプリング間隔に一致するため，誤差としては最小の値である。そのため，ここではシミュレーションが十分な精度で実施できたと結論づける。特に未転送データ量 *QueuedWrites* が 300MB 以上蓄積されている時刻（図 7.18 斜線部）では，17 時 40 分時点を除いて誤差が全く発生していないため，最大リカバリポイントを予測する手法としては十分な精度と結論づけて問題ない。

一方，未転送データ量が 300MB より少ない時間帯では，ほぼすべてのポイントで計算値と実測値の間に 5 分から 10 分の誤差が生じている。これはクラウドゲートウェイがデータ圧縮効果を得るなどの目的で，ある程度のデータ量がバッファに溜まるまで転送を抑えていた可能性が考えられる。こうした時間帯では一時保管時間 $D$ を長めに設定することで，その振る舞いを再現し，シミュレーションの精度を高められる。しかしながら，クラウドゲートウェイの実装に関わる仕様を利用者側では把握できないため，

本研究での検証は見送る。アップロードバッファのデータ蓄積量が少ない区間ではリカバリポイントも短時間にとどまるため，本研究の目的であるデータ転送保護性能を測る最大リカバリポイントの推定にあたっては問題にならないと考える。

(2) 適正システムサイズの導出

図 7.19 は，データ転送ネットワーク帯域を 4Mbps から 20Mbps に設定した 8 種類のシステム構成を想定し，それぞれについて性能シミュレーションを適用した結果を表している。(1)と同じく，バッファ一時保管時間 $D$ は 5 分，転送効率 $Transmission\_Efficiency$ は 85%に設定した。図 6.5 の想定どおり，データ転送に用いる回線帯域が広いほどアップロードバッファへのデータ蓄積量が減少し，リカバリポイントが短くなる。逆に，帯域が狭いほど未転送データの最大蓄積量が大きくなり，より大容量のストレージがアップロードバッファに必要となる。

6.3.3 節で定義したとおり，このケースで変動するコスト要因はデータ転送ネットワークの帯域 $Cost_{Network}$ とアップロードバッファのストレージ容量 $Cost_{UploadBufferCapacity}$ である。8 種類の想定のうち，RPO を達成し，かつこれらのコストの和 $Cost_{Variable}$ が最小となるものを適正システムサイズとして導出する。

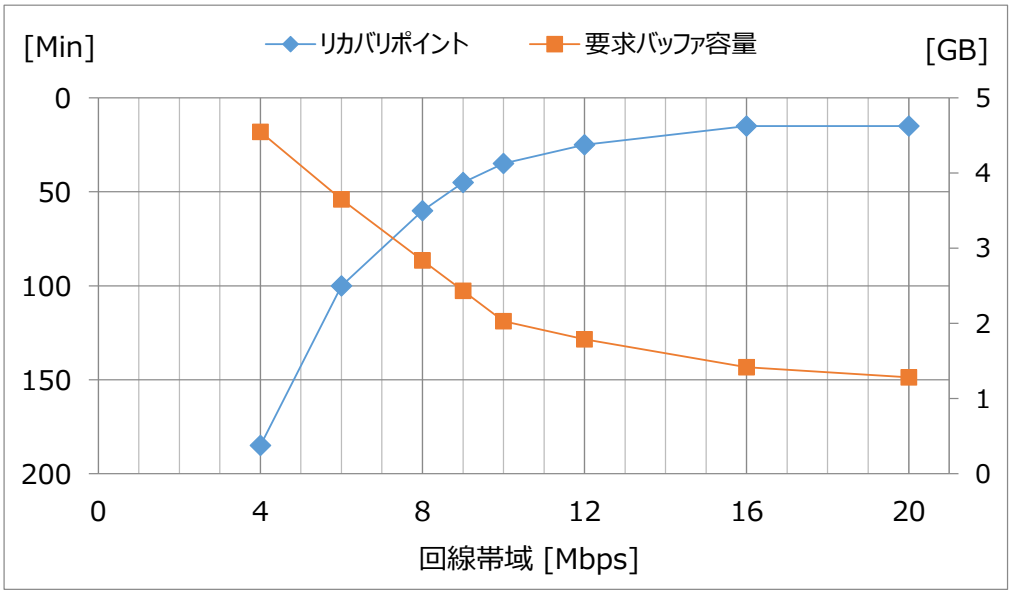



**図 7.19 シミュレーション結果比較**

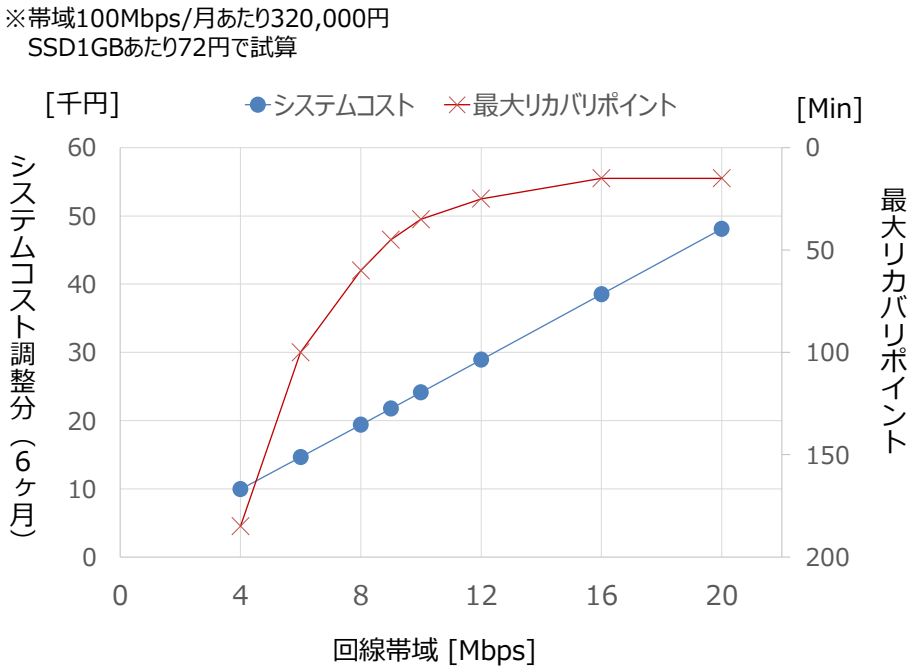



**図 7.20 システムコスト比較**

7.1.1 の試算と同様，コスト調整分の計算はシステム更改後 6 ヶ月分の費用とした。本試算においても，ストレージコストに対してネットワーク回線コストが支配的となり，回線帯域に比例してコストが増加する結果となった。したがって，ここでもリカバリポイントが RPO を達成する構成のうち，最も帯域の小さい構成を適正システムサイズとして導出すれば良い。

## 7.3. 考察

### 7.3.1. シミュレーション精度

前節では，クラウドで構築した実験環境を用いて技術の妥当性を評価した。本提案におけるサイジング方式の根拠となる未転送データ量に関しては，シミュレーションによる予測値と実験による測定値の誤差をピーク時点で 1.9%まで縮めることができた。これは時間に換算するとリカバリポイントの誤差が発生しない範囲であり，シミュレーシ

ョンとしては十分に高精度であると結論づけられる。

### 7.3.2. システムコスト削減効果

従来のウォーターフォール開発におけるシステム設計プロセスでは，事前に負荷の変動を予測することが困難であったため，設計段階で負荷のピーク値にあわせた量のシステムリソースを準備することが一般的であった。図 7.10 の例における書き込み負荷のピーク値は 858MB/min，これは 111.7 Mbps に相当する。図 7.21 に示すとおり，ピーク時点 111.7Mbps にあわせた設計におけるコストは 6 ヶ月あたり 2145 千円と見積もれる。ひとつの考察として，性能目標をピークにあわせるのではなく，RPO を 30 分と仮定すると，その達成に必要な回線帯域は 12Mbps（リカバリポイント＝25 分）であり，6 ヶ月あたりの支出は 230 千円となる。すなわちピークにあわせた構成に対し，本研究方式を適用することで，調整可能なコストを 89%削減できることがわかった。

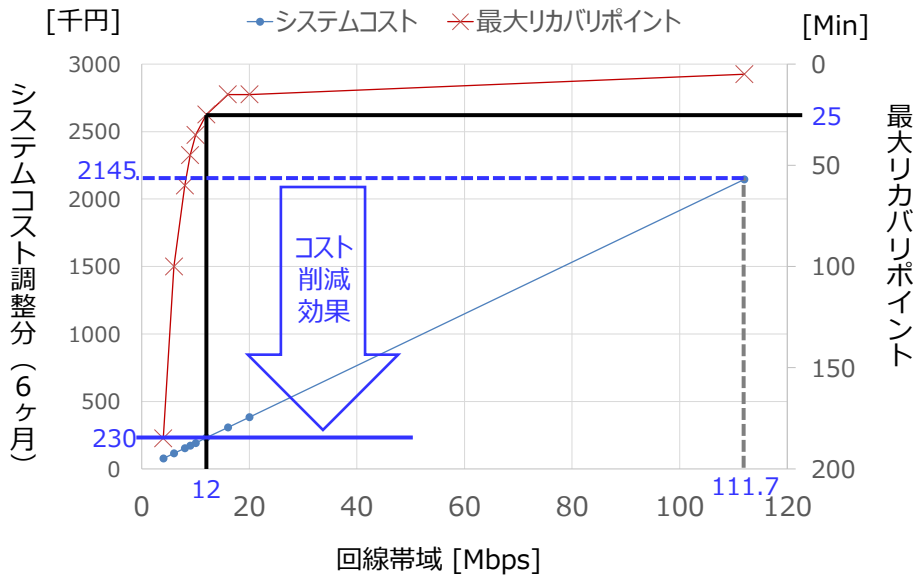



**図 7.21 コスト削減効果（RPO = 30min）**

### 7.3.3. 提案技術の汎用性について

図 7.10 に挙げた以外の書き込み負荷を対象とした性能評価およびシステムサイジング方式の検証結果を以下に示す。傾向が異なるいずれの負荷パターンについても，本研

究技術の適用によりリカバリポイントの算出とシステムサイジングを実行できる。これらに図 7.10 の書き込み負荷（パターン１）を加えた検証結果を表 7.3 にまとめる。

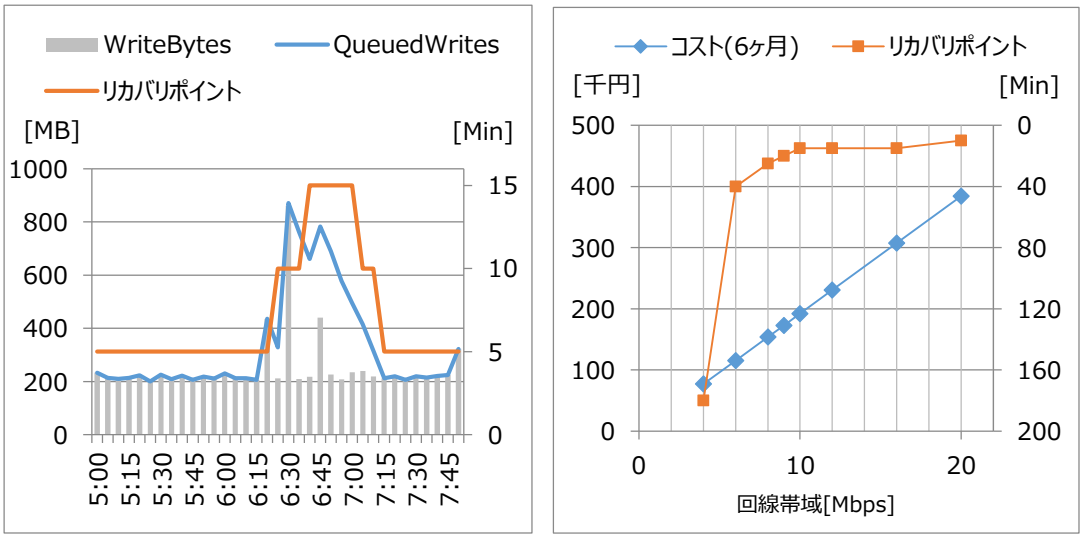



図 7.22 性能評価およびシステムサイジング試行結果（負荷パターン２）

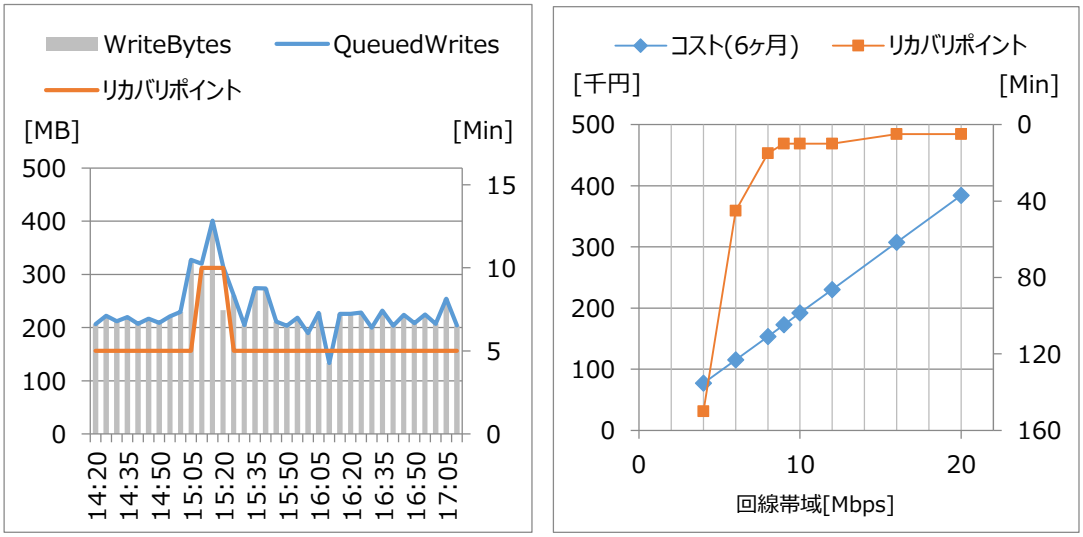



図 7.23 性能評価およびシステムサイジング試行結果（負荷パターン３）

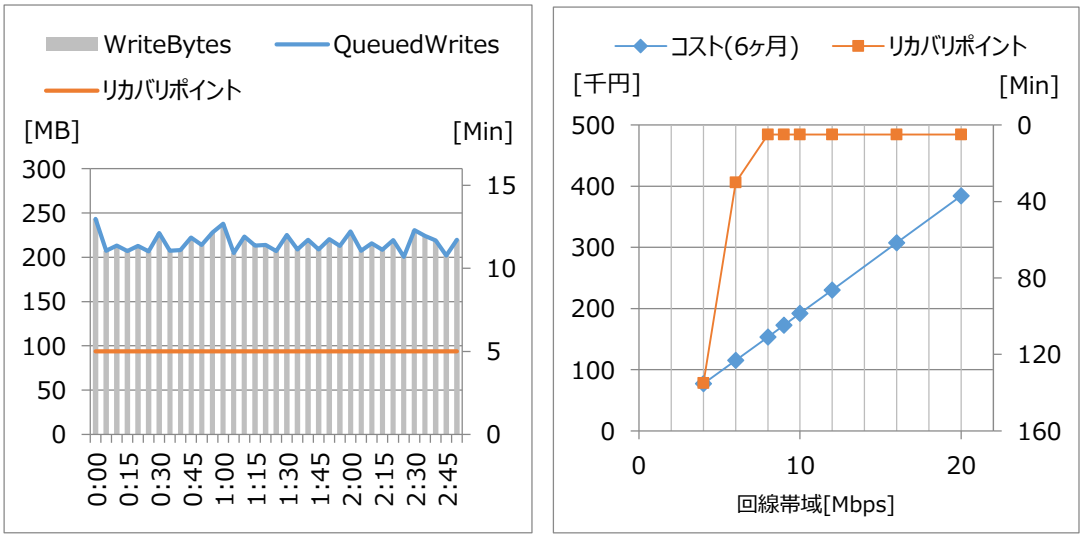



**図 7.24 性能評価およびシステムサイジング試行結果（負荷パターン４）**

**表 7.3 負荷パターンごとのコスト削減効果**

| # | 負荷パターンの特徴 | | 書き込み最大値(a)にあわせた性能設計 | | RPO にあわせた性能設計 | | | |
|---|---|---|---|---|---|---|---|---|
| | 書き込みデータ量最大値(a) (MB/min) | 標準偏差 | 回線帯域 (Mbps) | コスト調整分 (千円／６ヶ月) | RPO (min) | 回線帯域 Mbps | コスト調整分 (千円／６ヶ月) | コスト削減効果 (%) |
| 1 | 858 | 335 | 111.7 | 2145 | 30 | 12 | 230 | 89 |
| 2 | 631 | 112 | 82.3 | 1580 | 30 | 8 | 154 | 90 |
| 3 | 140 | 46 | 18.3 | 351 | 30 | 8 | 154 | 56 |
| 4 | 64 | 11 | 8.4 | 161 | 30 | 6 | 115 | 28 |

表 7.3 に挙げた通り，ピーク時点の書き込み負荷と，偏差が大きい負荷パターンにおいて，より大きなコスト削減効果が得られることがわかる。これはすなわち図 7.10，図 7.22 に挙げたように，ある時刻の書き込みが他の時間帯を大きく上回るようなピークを有するパターン，あるいは周期的に負荷が著しく増減するパターンなど，書き込みデータ量が大きく変動するケースにおいて，効果を得られることを意味している。逆に，図

7.24 に示した負荷の変動が小さいパターンでは大きな効果を得にくい。これは，図 5.4 のモデルにおいて，書き込み負荷の大きい場面でバッファにデータを滞留し，負荷の小さい状況でバッファのデータをネットワークに排出するゲートウェイの機構が有効に働くためには，負荷の変動が発生していることが前提となるためである。

### 7.3.4. さらなる精度向上に向けた技術課題

本研究で検証に至らなかった技術課題を以下に示す。これらの課題を解決することで，本方式の妥当性および計算精度を高められる可能性がある。

(1) データ転送遅延の実測と，計算による予測値との誤差

図 5.1 で述べたように，転送データにタイムスタンプを付与することで，転送遅延時間を測定することが可能となる。この実測値と，計算によるリカバリポイントの予測値の誤差を比較することで，本研究技術の精度を検証することが可能となる。

(2) 転送データ圧縮効果の影響

また，アップロードバッファに滞留するデータを圧縮することで，転送するデータ量を削減することができると考えられる。データ圧縮の影響調査も課題のひとつである

(3) 通信性能の予測

本評価のシミュレーションでは，データ転送性能の予測に課題を残した。6.3.2 節で述べたとおり，ローカルサイトからリモートサイトへデータを転送する通信回線の帯域に，転送効率パラメータをかけ合わせることで転送性能を再現する方式を採用したが，この転送効率を予測する計算方式の確立には至っていない。

また，データ転送をインターネット経由で行う場合には，その転送遅延時間などを事前の予備実験などで測定しておくことが望ましい。これらのデータ転送性能の予測手段については今後の課題とする。

# 8章　結論

## 8.1. 本研究の成果

(1) データ保護システム性能（リカバリポイント）評価方式の確立

データ保護システムの運用においては，できるだけ遅滞なくバックアップサイトへのコピーを実行することで，データ消失のリスクを抑えることができる。すなわち書き込み負荷に応じて変動するデータ転送の遅延に相当する同システムのリカバリポイントを常に監視し，性能目標である Recovery Point Objective（RPO）を達成しているかどうか，常に判定できることが望ましい。

データ転送遅延を測定するために，従来は公開されていない内部パラメータを用いる手法が提案されていた。これに対し，本研究では標準的な監視パラメータ（アップロードバッファに滞留する未転送データ量）を遅延時間に変換する計算式を定義した。すなわち，特定の製品やコピー機能に依存するのではなく，標準パラメータを入力とすることで，様々な方式に適用できる汎用的なリカバリポイント計算方式を確立した。本方式により，ストレージやデータベース，クラウドなどデータコピー方式が異なるシステムであっても，共通の計算式によりリカバリポイントを計測し，その性能を評価することが可能となった。

(2) 性能シミュレーションを通じたシステムサイジング方式の確立

データ保護システムの運用にあたっては，書き込みの負荷と性能，コストのバランスを定期的に検証して，システムサイズを適正化することが望ましい。そこで本研究では，書き込みデータ量の負荷変動にあわせて，システム性能（リカバリポイント）が性能目標値（RPO）を達成するようにシステムサイズを制御する方式を提案した。リカバリポイントの決定要因であるデータ転送遅延の原因には，コピー処理におけるバッファでの一時滞留時間や，データ転送ネットワークの性能上限などが挙げられる。このうち前者

はコピー方式の仕様に依存するためコントロールできないが, 後者のネットワーク性能は許容できるリスク（すなわち RPO）にあわせて調整可能である。そこで本研究では, ネットワーク性能上限を入力パラメータとして, リカバリポイントを予測する計算式を定義した。この計算式を用いたシミュレーションを通じて，RPO を達成し，かつネットワーク性能（すなわちネットワークコスト）が過剰でないシステム構成を導出することを可能とした。

実験の結果，負荷のピーク時点における誤差を 1.9%に抑えられたことで，同シミュレーションが十分に高精度であるとの結論に至った。さらに負荷のピークにあわせたシステム構成と比較して，あるリスク（RPO 30 分）を許容する構成では６ヶ月分のコストを 89%削減可能となる実験結果を得られた。

(3) クラウドを活用したデータ保護システム運用管理サイクルの確立

従来のストレージやデータベースが具備する非同期データコピー方式は, できるだけ遅滞なくデータを転送しようとするため，遅延時間もミリ秒オーダ程度を想定できた。その後クラウドが普及し, バックアップデータをクラウドストレージに保管するようになるが, そのデータ転送処理を担うゲートウェイサービスは分オーダの遅延を前提とする上，そのリスク管理に必要な管理機能が提供されないことが課題となった。そこで本研究では, クラウドの監視サービスでも供給される標準的な監視パラメータを用いたリカバリポイント予測方式を提案した。同方式適用により，システムを構成するリソース量を適正化する運用管理を可能とした。

データ保護システムの運用管理にあたっては，Plan（RPO が適正か）・Do（RPO を達成しているか）・Check（過剰なリソース量となっていないか）・Action（システムサイズの適正化）の各フェーズを実現するための技術が必要となる。本研究技術の適用により，これらの PDCA サイクルを確立し，リソースを適切に調整しながらコストを抑えることが可能となった。

## 8.2. 今後の展開

7.3 に述べたとおり，残された技術課題を解決することで，本研究の性能シミュレーションの妥当性を検証し，その精度をさらに高められる可能性がある。特にデータ転送遅延時間を実測し，予測値との誤差を測ることで，そのシミュレーション精度を検証することは重要である。

また，本提案方式はデータ保護に限らず，拠点間でデータを転送する様々なアプリケーションに応用可能である。一例として 3.2.4 に述べたとおり，これからは産業機器や自動車，監視カメラをはじめ様々な種類の IoT デバイスがネットワークに接続されるようになる。IoT デバイスの共通的な特徴は，それぞれが生成したデータをデータセンタに転送するように振る舞うことである。異常検知や危険予測といったリアルタイムの応答を要する IoT アプリケーションでは，データ転送遅延が重大な問題の原因となり得る。したがって，これらのアプリケーションが性能要件を達成するためには，データ転送性能の監視と性能の適正化が必要となる。本研究技術は，書き込みデータ量とシステムリソースのサイズからデータ転送性能を予測する汎用的な手法であり，またネットワークプロトコルへのタイムスタンプの追加といった特別な実装を必要としないため，データバックアップ以外の用途にも容易に適用できる。こうした IoT 分野への応用は今後の応用のひとつである。

加えて，本研究では性能シミュレーションにあたり，ストレージやクラウドのシステム仕様や振る舞いを十分に理解し，その知識に基づいてモデルを定義する演繹的な手法を採用した。一方で，3.5.3 に述べたように，システム稼働監視データを大量に蓄積できれば，機械学習などの帰納的な手法によってシステムモデルを開発できるようになる可能性がある。

# 謝辞

本研究の全過程を通じて，終始懇切丁寧なるご指導とご鞭撻，ならびに格別のご配慮を賜りました電気通信大学大学院情報システム学研究科情報ネットワークシステム学専攻の吉永努教授に深く感謝を申し上げます。論文審査をご快諾いただいた大学院情報システム学研究科の大森匡教授，田野俊一教授，大坐畠智准教授，策力木格准教授に深く感謝申し上げます。社会人の立場でありながら，研究室に受け入れてくださった吉永研究室の皆様に感謝を申し上げます。

筆者が電気通信大学大学院情報システム学研究科情報システム学専攻博士後期課程に在学することへのご配慮と援助を賜りました株式会社日立製作所研究開発グループ鈴木教洋博士，岩嵜正明博士，技術戦略室 赤津雅晴博士，テクノロジーイノベーション統括本部 矢川雄一氏に感謝致します。

本研究の機会と大学院博士後期課程に進学する機会を与えて頂くとともに，格別のご指導をいただきました株式会社日立製作所研究開発グループ基礎研究センタ 水野弘之博士，未来投資本部 中屋雄一郎博士に感謝申し上げます。

本研究の実験の機会と環境を与えていただき，多数のご助言を賜りましたサービスプラットフォーム事業本部 中村輝雄氏，同 IoT・クラウドサービス事業部 吉田高明氏，山本祐輔氏に感謝致します。また，論文をご指導いただいた研究開発グループテクノロジーイノベーション統括本部デジタルテクノロジーイノベーションセンタ 保田淑子博士，入社直後から長きにわたり共に研究を行い，懇切丁寧にご指導いただいた社会イノベーション協創統括本部山本政行氏，本研究に共に取り組んだ市川直子氏に心より感謝申し上げます。

そして日々の活動を支えて下さった秘書の皆様，格別なるご指導とご配慮を賜りました株式会社日立製作所研究開発グループの皆様に心から御礼申し上げます。

学位取得に向けて応援してくれた両親と姉に感謝いたします。最後に，本論文の執筆にあたりすべてのサポートをしてくださった妻と，いつも笑顔と勇気と元気をくれた子どもたちに心から感謝します。

# 参考文献


[1] Gantz, J., Reinsel, D., “THE DIGITAL UNIVERSE IN 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East,” IDC, 2012.

[2] Reinsel, D., Gantz, J., Rydning, J., “Data Age 2025: The Evolution of Data to Life-Critical, Don't Focus on Big Data; Focus on the Data That's Big,” IDC, 2017.

[3] “The Zettabyte Era: Trends and Analysis,” Cisco, 2017.

[4] Gupta, C., “Collaborative Creation with Customers for Predictive Maintenance Solutions on Hitachi IoT Platform,” Hitachi Review Vol.65, No.9, 2016. pp.403-409

[5] Takeda, N., Kita, Y., Nakagawa, H., Suzuki, H., “Using Operation Information in Reliability and Maintenance: Analytics ofr the IoT Era,” Hitachi Review Vol.65, No.9, 2016. pp.450-455

[6] Vennelakanti, R., “Winning in Oil and Gas with Big Data Analytics,” Hitachi Review Vol.65, No.2, 2016. pp.884-888

[7] LaValle, S., Hopkins, M., Lesser, E., et al., “Analytics: The new path to value,” MIT Sloan Management Review, 2010.

[8] Manyika, J., Chui, M., Brown, B., et al., “Big Data: The next fronier for innovation, competition, and productivity,” McKinsey Global Institute, 2011.

[9] Patterson, David A., “A Simple Way to Estimate the Cost of Downtime,” USENIX 16th System Administrators Conference (LISA), 2002. pp.185-188

[10] Toigo, Jon W., Disaster Recovery Planning, Prentice Hall, 2003.

[11] Keeton, K., Santos,C., Beyer,d., et al., “Designing for disasters,” USENIX conference File and Storage Technologies (FAST), 2004. pp.59-62

[12] Rudolph, C. G., “Business Continuation Planning/Disaster Recovery: A Marketing Perspective,” IEEE Communication Magazine, 1990. vol.28, no. 6, pp.25-28, doi:10.1109/35.56224

[13] Hitachi Vantara, “Simplify Data Protection and Recovery, Use the Right Tool for Each Job, but Manage Them From One Place,” 2017.

[14] Patterson, D. A., Gibson, G., Katz, R. H., “A Case for Redundant Arrays of Inexpensive Disks (RAID),” ACM SIGMOD International Conference on Management of Data, 1988. pp.109-116

[15] 大和純一, 管真樹, 菊地芳秀, “広域災害に対するストレージによるデータ保護,” 電子情報通信学会誌 89 巻 9 号, 2006. pp.801-805

[16] Quarantelli,E., “The Disaster Recovery Process: What we know and we do not know from research,” Disaster Research Center, 1999.

[17] 江丸裕教, 高井晶彰, 原純一, “ディザスタリカバリにおける非同期リモートコピーのリカバリポイント監視方式,” 情報処理学会研究報告, 2010. vol.2010-EVA-31, no.1

[18] Alhazmi, O. H., Malaiya, Y. K., “Evaluating Disaster Recovery Plans Using the Cloud,” Reliability and Maintenability Symposium (RAMS), IEEE Proceedings-Annual, 2013.

[19] Armbrust, M., Fox, A., Griffith, R., et al., “Above the Clouds: A Berkeley View of Cloud Computing,” Electrical Engineering and Computer Sciences, University of California at Berkeley, 2009.

[20] IBM White Paper, “Virtualizing disaster recovery using cloud computing,” IBM Global Technology Services, 2012.

[21] Alhazmi,O. H., Malaiya,Y. K., “Assessing Disaster Recovery Alternatives: Onsite, Colocation or Cloud,” IEEE 23rd International Symposium on Software Reliability Engineering Workshops, 2012. pp.19-20

[22] Wood, T., Cecchet, E., Ramakrishnan, K. K., “Disaster Recovery as a Cloud Service: Economic Benefits and Deployment Challenges,” 2nd USENIX Workshop on Hot Topics in Cloud Computing, 2010.

[23] Javaraiah, V., “Backup for Cloud and Disaster Recovery for Consumers and SMBs,” IEEE 5th International Conference on Advanced Networks and Telecommunication Systems, 2011.

[24] Wiboonratr, M., Kosavisutte, K., “Optimal Strategic Decision for Disaster Revovery,” International Journal of Management Science and Engineering Management, 2009. vol.4, no.4, pp.260-269

[25] Gopisetty, S., Butler, E., Jaquet, S., et al., “Automated planners for storage provisioning and disaster recovery,” IBM Journal of Research and Development Vol.52, 2008. pp.353-365, doi:10.1147/rd.524.0353, ISSN:0018-8646

[26] Nayak, T., Routray, R., Singh, A., et al., “End-to-end Disaster Recovery Planning: From Art to Science,” IEEE/IFIP Network Operations and Management Symposium (NOMS), 2010. pp.357-364, doi:10.1109/NOMS.2010.5488491, ISBN:978-1-4244-5366-5

[27] Ji, M., Veitch. A., Wilkes, J., “Seneca: Remote Mirroing Done Write,” Proceedings of USENIX Technical Conference, 2003. pp.253-268

[28] Wood, T., Lagar-Cavilla, H. A., Ramakrishnan, K. K., et al., “PipeCloud: Using Causality to Overcome Speed of Light Delays in Cloud-Based Disaster Recovery,” Proceedings of the 2nd ACM Symposium on Cloud Computing Article No. 17

[29] Cully, B., Lefebvre, G., Meyer, D., “Remus: High Availability via Asynchronous Virtual Machine Replication,” 5th USENIX Symposium on Networked Systems Design and Implementation, 2008. pp.161-174

[30] Caraman, M. C., Moraru, S. A., Dan, S., Grama, C., “Continuous Disaster Tolerance in the IaaS Clouds,” 13th IEEE International Conference on Optimization of Electrical and Electronic Equipment (OPTIM), 2012. pp.1226-1232, doi:10.1109/OPTIM.2012.6231987, ISBN:978-1-4673-1653-8

[31] 田村芳明, 柳澤佳里, 佐藤孝治, 盛合敏, “Kemari: 仮想マシン間の同期による耐故障クラスタリング,” 情報処理学会論文誌 コンピューティングシステム, 2010. vol.3, no.1, pp.13-24

[32] Zhu, J., Jiang, Z., Xiao, Z., Li, X., “Optimizing the Performance of Virutal Machine Synchronization for Fault Tolerance,” IEEE Transactions on Computers, 2011. vol.60, no.12, pp.1718-1729

[33] Rajagopalan, S., Cully, B., O'Connor, R., Warfield, A., “SecondSite: Disaster Tolerance as a Service,” VEE '12 Proceedings of the 8th ACM SIGPLAN/SIGOPS Conference on Virtual Execution Environments, 2012, vol.47, Issue 7, pp.97-108, doi:10.1145/2151024.2151039, ISBN:978-1-4503-1176-2

[34] Ueno, Y., Miyaho, N., Suzuki, S., Ichihara, K., “Performance Evaluation of a Disaster Recovery System and Practical Network Applications in Cloud Computing Environments,” International Journal on Advances in Networks and Services, 2011. vol.4, pp.130-137

[35] Grolinger, K., Mezghani, E., Capretz, M.A.M., Exposito, E., “Knowledge as a Service Framework for Disaster Data Management,” IEEE 22nd International WETICE Conference, Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2013, pp.313-318, doi:10.1109/WETICE.2013.48

[36] Moore, G., “Systems of Engagement and The Future of Enterprise IT, A Sea Change in Enterprise IT,” 2011.

[37] Chen, W., Kamath, R., Kelly, A., et al., “Systems of Insight for Digital Transformation,” IBM RedBooks, 2015. http://www.redbooks.ibm.com/redbooks/pdfs/sg248293.pdf

[38] Gens, F., “White Paper, The 3rd Platform: Enabling Digital Transformation,” IDC, 2013.

[39] Winnard, K., Fadel, L., Hunt, R., et al., “IBM Mainframe Bits: Understanding the Platform Hardware,” IBM RedBooks, 2016.
http://www.redbooks.ibm.com/redpapers/pdfs/redp5346.pdf

[40] 山本彰, 松並直人, 岩嵜正明, 吉田浩, 大枝高, “ストレージサブシステム,” 電子情報通信学会知識ベース，知識の森, 2011. ８群２篇５章

[41] VMware, “Storage Protocol Comparison White Paper,” 2012.

[42] 日立製作所, “ホワイトペーパー，日立サーバ論理分割機構「Virtage」と Intel Virtualization Technology による Nested Virtualization（VMM on LPAR）の実現,” 2014.

[43] VMware, “White Paper, VMware Infrastructure Architecture Overview,” 2006.

[44] 鳥谷部昭寛, “デジタル時代のクラウド活用戦略 マルチクラウド化の潮流,” 知的資産創造, 2017. pp.20-41

[45] Leffingwel, D., “Whitepaper, Mastering the Iteration: An Agile White Paper,” Rally Software Development Corporation, 2007.

[46] 情報処理推進機構, “「情報システム運用時の定量的信頼性向上方法」に関する調査報告書,” 2015.
https://www.ipa.go.jp/files/000045091.pdf

[47] 八木隆, 荻野美穂, 松本嘉夫, 藤本昌代, “「ITIL」導入のプロセスと効果を知る,” Computerworld, 2004.
http://www.hitachi.co.jp/products/it/itil/information/pdf/cw_itil.pdf
[48] Amazon Web Services, “Amazon CloudWatch User Guide,” 2018.
[49] 日立製作所, “特集 2016 年 日立技術の展望 IT ソリューション・サービス,” 日立評論 2016 年１・２月合併号, p. 42, 2016. pp.47-53
[50] Buffington, J., Yuen, E., “Data Protection Should Be Part of Your Systems Management Strategy,” 2018.
[51] 株式会社情報通信総合研究所, “ビッグデータの流通量の推計及びビッグデータの活用実態に関する調査研究報告書,” 2015.
[52] Cisco, “Cisco Global Cloud Index: Forecast and Methodology, 2016-2021,” Cisco Public, 2018.
[53] 総務省, “第１章第３節ビッグデータの活用が促す成長の可能性,” 平成 25 年度版 情報通信白書, 2013. pp.143-179
[54] 総務省, “第３章データが切り拓く未来社会,” 平成 26 年度版 情報通信白書, 2014. pp.100-137
[55] 矢野和男, “AI で予測不能な時代に挑む,” 日立評論, 2016. pp.12-32
[56] 久間和生, “Society 5.0 実現に向けて,” 内閣府総合科学技術・イノベーション会議, 2016.
http://www8.cao.go.jp/cstp/tyousakai/juyoukadai/infra_fukkou/12kai/sanko2.pdf
[57] 内閣官房日本経済再生総合事務局, “未来投資戦略 2017 Society5.0 の実現に向けた改革,” 2017.
http://www5.cao.go.jp/keizai-shimon/kaigi/minutes/2017/0609/shiryo_07.pdf
[58] 日立製作所, “Society 5.0 実現に向けた日立の取り組み”.
http://www.hitachi.co.jp/products/social/society5/download/Society_5_0.pdf
[59] 内閣府, “事業継続ガイドラインーあらゆる危機的事象を乗り越えるための戦略と対応ー,” 2013.
http://www.bousai.go.jp/kyoiku/kigyou/pdf/guideline03.pdf

[60] Hitachi Vantara, “Hitachi Vantara's Approach to 3-Data-Center Business Continuity and Disaster Recovery,” 2017.

[61] Rebah, B. H., Sta, B. H., “Disaster Recovery as a Service: A Disaster Recovery Plan in the Cloud for SMEs,” Global Summit on Computer & Information Technology (GSCIT), 2016. pp.32-37

[62] Mary, A., Chitra, K., “Study on Disaster Recovery in Cloud Environment,” World Congress on Computing and Communication Technologies (WCCCT), 2017. pp.165-167

[63] Alshammari, M. M., Alwan, A. A., Nordin, A., Al-Shaikhli, I. F., “Disaster Recovery in Single-Cloud and Multi-Cloud Environments: Issues and Challenges,” 4th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS), 2017.

[64] 宮崎淳子, 齋藤邦夫, 手塚大, 中村隆喜, 村岡裕明, “大規模災害発生時での高可用ストレージ基盤の耐災害性実証実験,” 平成 28 年度電気関係学会東北支部連合大会, 2016.

[65] Souza Couto, R. D., Secci, S., Campista, M.E.M. et al., “Network Design Requirements for Disaster Resilience in IaaS Clouds,” IEEE Communications Magazine, October 2014. vol.52, issue 10, pp.52-58, doi:10.1109/MCOM.2014.6917402, ISSN:0163-6804

[66] Yang, C., Huang, C., Kao, Y., Tasi, Y., “Disaster Recovery SIte Evaluations and Selections for Information Systems of Academic Big Data,” Eurasia Journal of Mathematics, Science and Technology Education, 2017. pp.4553-4589

[67] Schulman, R. R., “Disaster Recovery Issues and Solutions, A White Paper,” Hitachi Data Systems, 2003.

[68] Hitachi Vantara, “Data Protection: Downtime Is Money,” 2018.

[69] 加倉井宏一, 荻田光一郎, “災害対策システムのリニューアルにおける現実的災害対策レベルの評価,” 情報処理学会研究報告, 2004.

[70] Fegreus, J., “Hitachi Data Instance Manager Protects Active VM Applications in Real Time Without Backup Degradation,” openBench Labs, 2013.

[71] Oracle, “Oracle ホワイト・ペーパー，Oracle Active Data Guard，リアルタイム・データ保護と可用性,” 2015.

[72] Patterson, H., Manley, S., Federwisch, M., et al., “SnapMirror: File-System-Based Asynchronous Mirroring for Disaster Recovery,” USENIX conference File and Storage Technologies, 2002. No.9

[73] Shilane, P., Huang, M., Wallace, G., Hsu, W., “WAN Optimized Replication of Backup Datasets Using Stream-Informed Delta Compression,”ACM Transactions on Storage (TOS), 2012, vol.8, issue 4, No.13, doi:10.1145/2385603.2385606

[74] Khoshkholghi, M. A., Abdullah, A., Latip, R., et al., “Disaster Recovery in Cloud Computing: A Survey,” Computer and Information Science, 2014, vol.7, no.4, pp.39-54, ISSN:1913-8989

[75] Bajpai, A., Rana, P., Maitrey, S., “Remote Mirroring: A Disaster Recovery Technique in Cloud Computing,” IJARSE, International Journal of Advance Research in Science and Engineering, vol.2, issue 8, 2013. pp.166-175 ISSN:2319-8354

[76] Chang, V., “Towards a Big Data System Disaster Recovery in a Private Cloud,” Ad Hoc Networks, 2015. pp. 65-82

[77] Amazon Web Services, “AWS Storage Gateway User Guide,” 2018. https://docs.aws.amazon.com/storagegateway/latest/userguide/storagegateway-ug.pdf

[78] Al-Sharidah, A. H., Al-Essa, H.A., “Toward Cost Effective and Optimal Selection of IT Disaster Recovery Cloud Solution,” Computer Science and Electronic Engineering (CEEC), 2017. pp.43-47

## 関連論文の印刷公表の方法および時期

### 論文誌（査読有り）

[79] 田口雄一，山本政行，“ディザスタリカバリに向けた非同期リモートコピー構成資源算出方式,” 情報処理学会論文誌コンピューティングシステム（ACS）７巻２号, 2014, pp. 1-10
※５章～７章 ストレージシステムによる方式評価技術に関連

[80] Taguchi, Y., Yoshinaga, T., “Assessment and Simulation of System Performance of a Cloud-based Data-protection System”, Journal of Information Processing Vol.26, 2018, pp. 687-695
※５章～７章 クラウドサービスによる方式評価技術に関連

### 国際会議（査読あり）

[81] Taguchi, Y., Yoshinaga, T., “System Resource Management to Control the Risk of Data-Loss in Cloud-based Disaster Recovery”, 42nd IEEE International Conference on Computer Software & Applications, 2018, pp 210-215

### 研究会・シンポジウム発表（査読あり）

[1] 田口雄一，市川直子，山本政行，“ディザスタリカバリに向けた非同期リモートコピー構成資源算出方式,” 情報処理学会第 25 回コンピュータシステム・シンポジウム, 2013, pp. 45-53

[2] Taguchi, Y., Yoshinaga, T.,“System Performance Assessment and Sizing for Cloud-

based Data Backup,” 情報処理学会第 29 回コンピュータシステム・シンポジウム, 2017, pp. 41-48

## 論文誌（査読無し）

[1] 花岡誠之，田口雄一,中村友洋，加藤博光，鍛忠司，小味弘典，森脇紀彦，小日向宣昭，Ken Wood，橋本哲也，高村祐史，“社会イノベーション事業を広げる IoT プラットフォーム,” 日立評論イノベイティブ R&D レポート, 2016，pp. 52-56

## 研究会発表（査読無し）

[1] 丸山直子, 古橋亮慈, 田口雄一, 山本政行 , 兼田泰典,“ディザスタリカバリにおける運用設計の簡略化に向けた検討,” 情報処理学会 第 69 回全国大会講演論文集, 69 巻, 4 号, 2007, pp. 4.403-4.404

[2] 丸山直子, 田口雄一 , 山本政行, “ディザスタリカバリシステムにおけるストレージリモートコピー構成評価モデルの提案,” 情報処理学会 第 70 回全国大会講演論文集, 70 巻 4 号, 2008, pp. 4.539-4.540.

[3] 水野潤, 田中徹, 田口雄一, 山本政行, 佐藤雅英, 兼田泰典, “大規模ストレージシステム向けのボリューム設計方法の提案,” 情報処理学会 FIT2007（第６回情報科学技術フォーラム）, 2007, pp. 3-4

[4] 阿久根憲, 木下順史 , 田口雄一, “異種クラウド間接続のためのオーバレイネットワーク方式の提案,” 電子情報通信学会技術研究報告，114 巻，139 号, 2014, pp. 77-81

[5] 田中徹, 中嶋法子, 田口雄一, 田村賢司, “ストレージシステムにおける負荷傾向予測方式の提案,” 情報処理学会 FIT2009（第８回情報科学技術フォーラム），８巻４号, 2009, pp. 535-536