情報処理学会論文誌 Vol.53 No.2 672–680 (Feb. 2012)

最短時間でサービス回復を可能とする障害復旧フローの提案と評価

長野 伸一[1,a)] 谷口 友宏[1] 谷 洋介[1] 葛西 圭祐[1] 三堀 英彦[1]

受付日 2011年6月14日, 採録日 2011年11月7日

概要：インフラに利用されるような高い信頼性が要求されるコンピュータシステムは，そのための機能を具備し，手厚い検証を行うが，加えて運用中に遭遇する不測の事態にも迅速に対応できる業務フローが必要で，特に，どのような障害復旧方法を施すかが重要である．本論文では，最短時間で障害を復旧させるフローの作成方法を提案する．提案する作成方法は，コンピュータアーキテクチャの知識と対象システムの運用方針をもとに，動的計画法を利用して，複数の障害復旧方法を選択し，実施順序を決める．用いる復旧方法は，予備系への切替えや再起動など原因究明を必要としない復旧方法からなるため，早期の障害復旧を可能とする．また，当該手法を商用システムの障害復旧フローの作成に適用し，その実用性を運用者と議論し，導入した経験を報告する．

キーワード：障害復旧，最短時間，動的計画法

Proposal and Evaluation How to Make the Recovery Flow for Failures within the Minimum Time

Shin'ichi Nagano[1,a)] Tomohiro Taniguchi[1] Yosuke Tani[1]
Keisuke Kasai[1] Hidehiko Mitsubori[1]

Received: June 14, 2011, Accepted: November 7, 2011

Abstract: The computer system required high reliability such as infrastructure, is equipped with the functions to do so and verified them deeply and carefully. In addition, it needs the action flow which makes rapidly to manage unexpected emergency situation. Especially, it is important which set of methods should be tried in the flow. We propose how to make the recovery flow for failures within the minimum time. Our proposed method has feature to select and place means for failures in a line, by using the dynamic programming under the knowledge of computer architecture and the operational policy of the target system. Candidates of recovery method such as switching to standby or restart, does not need to explore causes of failures and therefore our proposed recovery flow can recovery the service providing quickly. We also report our experience to be applied for the communication system and the discussion with operators for it.

Keywords: disaster recovery, minimum time, dynamic programming

1. はじめに

社会生活を支えるインフラを構成しているコンピュータシステム（以後，システムと略す）は，24時間365日継続したサービス提供を要求される．このため，これらのシステム開発にあたっては，冗長化など [1], [2], [3], [4] の障害に備えた機能を具備させ，加えて手厚く検証 [5] することで，高い信頼性を実現する．

しかしながら，システムの運用中には想定外の要因でサービス提供に支障をきたすこと（以後，サービス影響と略す）があり，上述のような開発時の対応のみでは不十分である．たとえば，その要因には，開発中に排除できなかったソフトウェアバグや保守作業中の人為ミスなど（以

[1] 東日本電信電話株式会社
NIPPON TELEGRAPH AND TELEPHONE EAST Corporation, Shinjuku, Tokyo 163–8019, Japan
[a)] s.nagano@east.ntt.co.jp

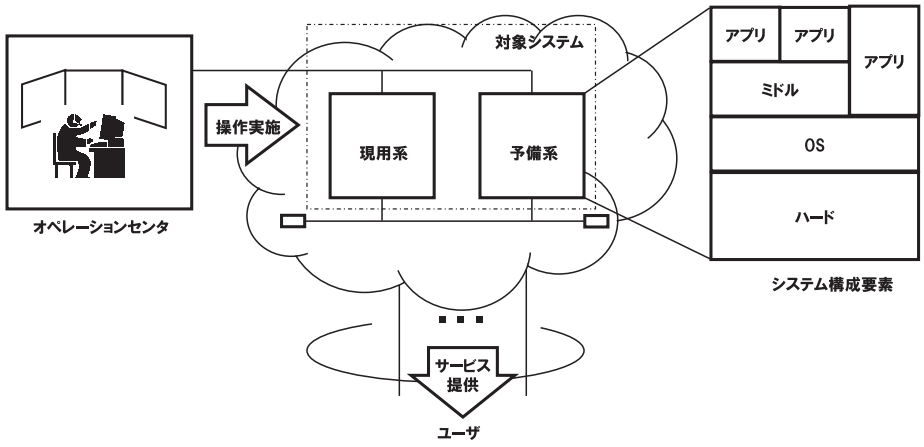


図 1 システム構成図
Fig. 1 A general system configuration.

後，障害と略す）などがある．このような障害を想定して，早期にサービス回復を図る運用フローを用意しておく必要があり，なかでも障害復旧が重要な課題である．

最も確実な障害復旧方法は，原因究明後にそれを除去する方法であるが，時間を要するうえ，必ずしも原因が判明するとは限らないという欠点がある．一方で，原因究明に頼らない方法として，冗長系への切替え，システム全体もしくは一部の立ち上げ直し（以後，再起動と呼ぶ），バックアップを用いた再起動などがあり，経験的に有効であることが知られている．これらの復旧方法は，必ずしも障害の原因を除去する根本的な対処になるとは限らないが，直面している障害を回避し，サービス回復を図るには有効である．

ただし，概してその種類は複数あり，障害原因によって，その効果が異なるため，最適な選択は容易ではない．それでも障害発生時には早期のサービス回復を強く要求されるため，異なる種類の再起動を順に試し，いたずらに時間を浪費する場合がある．したがって，適切な障害復旧方法の組合せを選択し，実施順を規定した障害復旧フローが強く望まれている．

そこで，本論文では，最短時間でサービス回復を可能とする障害復旧フローの作成方法を提案する．提案する方法は，当該システムが備えている，原因追求に依存しない障害復旧方法の最適な組合せを選択しつつ，対象システムに関する運用方針を盛り込み，フロー化できることが特徴である．また，障害復旧方法の最適な組合せに対しては，動的計画法 [6] を利用していることも特徴である．

まずは，2 章で前提とするシステム構成とその特徴を記し，我々の経験に基づき，そこで発生する障害と有効な障害復旧方法を示す．3 章で提案する障害復旧フローの作成方法を検討するうえでの要求条件を明示する．4 章で障害復旧フローを障害復旧方法の選択と実施順の決定の 2 つの課題に分け，前者を定式化し，動的計画法を利用して課題解決を図ることを示したうえで，提案する障害復旧フローの作成方法を述べる．5 章で商用システムに適用した評価結果を示す．最後に 6 章でまとめる．

2. 障害と従来の障害復旧方法

2.1 システム構成とその要素

インフラを構成するシステムでは，サービス制御，ハードウェア制御，データ管理など様々な処理を行うが，いずれもユーザ[*1]からの利用要求に応じて処理（サービス提供）するシステムに単純化することができる．ただし，その規模は概して大きく，同時に多数のユーザからの利用要求を処理する．

近年，これらのシステムには UNIX や Linux を搭載した汎用サーバが利用され，OS・ミドルウェア・アプリケーションなどのソフトウェアが階層構造をとることが一般的である（これらをシステム構成要素と呼ぶ）．また，信頼性を向上させるため，システム自体は冗長化されている．

このような構成を前提にした障害復旧方法には，障害系を切り離し，冗長系を運用系に遷移させる（以後，系切替えと呼ぶ）方法や，電源の OFF/ON，各種再起動，バックアップファイルを用いた再インストールがあり，これらはオペレーションセンタからオペレーションサポートシステム（OSS）を通して操作することができる．上述のイメージを**図 1** に示す．

2.2 障害原因と復旧方法

障害とその復旧方法の検討は様々に試みられている [7], [8], [9], [10], [11], [12] ものの，システムの特徴により異なるため，汎用的なものはない．そこで，筆者らの経験に基づいて，インフラで発生する障害の主要な原因とその復旧方法について以下にまとめるとともに，一覧にしたものを**表 1** に示す．

[*1] ユーザ：人間だけでなく，機械やソフトウェアも含む．

表 1 障害原因とその復旧方法
Table 1 Malfunction causes and recovery methods.

通番	障害原因	復旧方法	備考
1	ハード故障・通信障害	系切り替え	
		装置交換	復旧時間が長い
2	過負荷	利用制限・中止	
		再起動	
3	設定誤り・操作誤り	切り戻し	
		再インストール	サービス影響が大きい
4	データ破壊	リストア	直近に追加したデータの消失の可能性有り
		再起動	
5	ソフトウェアバグ	パッチ，ファイル更新	復旧時間が長い
		ライブパッチ	復旧時間は長いが，サービス中断無く実施可
		再起動	

(1) ハード故障，通信障害

物理的な故障．その復旧方法は系切替え [13], [14] もしくは装置交換である．特に，系切替えは短時間で実施できるため，直面している障害を回避して，サービス回復を図るのに有効である．当該方法は，ハードウェアの状態を変更する復旧方法である．

(2) 過負荷

想定をはるかに上回る量の利用要求があり，応答時間の劣化あるいは利用要求の破棄が生じる障害．復旧には利用要求の受け付けを制限あるいは中止し，受付済みの利用要求を完了させる（想定内の量に変化させる）が，受付済みの利用要求を完了させるために多大な時間を要する場合には，再起動 [15], [16], [17] などで途中放棄させることもある．

(3) 設定誤り・操作誤り

ハードウェアの変更やソフトウェアの更新，システムの動作条件の設定変更などの運用作業において，設定誤りや操作誤りにより，正常運行を阻害する状態にシステムを遷移させてしまった障害である．この障害に対する復旧方法は，設定や操作を正常な状態に戻すことである．すなわち，障害が発生した直前の設定や操作を疑い，設定変更前あるいは操作前のシステムの状態に戻す（切り戻し）ことである．切り戻しが困難な場合，再起動やバックアップファイルによる再インストールで復旧させる．

(4) データ破壊

ハードディスクやメモリ上のデータが，それを扱うソフトウェアの想定外の状態となり，誤作動を招く障害である．データ修復により復旧することができるため，その復旧方法は，バックアップデータを用いたリストアのほか，自システム内で生成される場合には再起動により修復することもある [18], [19].

(5) ソフトウェアバグ

ソフトウェアの誤り．それを修正することでサービス回復するため，その根本的な復旧方法はプログラム修正である．ところが，サービスを提供する環境への反映にはパッチやファイル更新が必要で，サービス提供の中断を必要とする場合がある．なかには，サービス提供を中断することなく，パッチ適用を可能とする方式もある [20] が，いずれにしても原因追求・対処プログラムの作成・検証の作業が必要となる．一方，ソフトウェアバグの要因に着目すると，プログラムロジックの問題，あるいはシステムの状態，およびその両方の組合せに依存する．プログラムロジックのみに起因するソフトウェアバグには汎用的な復旧方法はなく，個々に応じた対処が必要であるが，システム状態を障害の発生条件に含む場合，再起動によりシステム状態が変更され，障害発生を回避することができ，復旧方法として有効である．

以上のように，上述した復旧方法はシステムの状態を変化させることで，障害から復旧を図る点で共通している．このとき，遷移先は正常に稼動していた実績のある状態であり，システムの初期状態（立ち上げ直後）が多い．そこで，以後，障害復旧として実績のある状態に遷移させることを初期化，初期化するシステム構成要素の集合を初期化範囲と呼ぶこととする．

3. 障害復旧フローとその要求条件

システムの運用中に障害が発生した場合，根本的な障害の原因除去もさることながら，サービス回復が重要で，それを優先する．そのため，系切替え，再起動，再インストールなどの復旧方法が有利である．しかしながら，それが有効な障害の症状と対応していないため，障害原因が判明していない段階での最適な選択は困難である．さらに，再起動や再インストールはそれ自体にも様々な種類があり，いっそう復旧方法の選択を複雑にしている．

そこで，複数の復旧方法を順に試し，サービス回復を図る方法が考えられるが，無作為の試行は，いたずらにサービス回復までの時間を長引かせるだけである．また，復旧方法の中にはサービス提供の中断をともなうものもあり，

障害による被害を拡大させる懸念もある．

一方，障害に対する効果に着目すると，復旧方法ごとに初期化するシステム構成要素が異なるため，最適な組合せの選択が課題となる．たとえば，OS の再起動にはアプリケーションの再起動もともなうので，アプリケーション再起動後の OS 再起動は意味がない．

さらに，障害復旧フローの作成にあたっては，上述の議論以外にも，運用上の観点から考慮すべき項目がある．以下に，我々がこれまでインフラを運用してきた経験から，障害復旧フローに対する要求条件を示す．

(1) 回復時間の推定

障害復旧に際して，実施時間が推定できること．これにより，サービス回復時刻を想定することができ，利用者への対応が可能となる．

(2) 影響範囲の明確化

復旧方法の中には，サービス影響をともなうものもある．その場合でも，影響内容とその時間が明確であれば，実施後のユーザ対応が可能となる．当該復旧方法を実施した場合のサービス影響とその範囲が明確であることが必要である．

(3) 運用ポリシへの準拠

システムを運用するにあたり，当該システムの特性・ユーザ種類（人間/機械/ソフトウェアなど）・ユーザ規模・運用時間など様々な条件から，運用方針を規定している．これを運用ポリシと呼ぶ．

運用ポリシには，運用に必要なすべてを規定しており，障害に関する規定も含まれる．たとえば，障害の定義・障害規模に応じた報告先・サービス影響のある復旧方法を実施する際の判断基準などがある．

障害復旧フローの作成にあたっては，運用ポリシに則していることが必須である．

4. 障害復旧フローの作成方法

本章で提案する障害復旧フローの作成方法を述べる．2 章および 3 章の議論から，提案する障害復旧フローの概要は次のとおりである．

- 運用中に遭遇した障害に対して，一刻も早いサービス回復を行うために 2.2 節で述べた復旧方法を用いる．
- 故障部位は判明していないので，複数の復旧方法を順に実施し，つど，サービス回復を確認する．
- 復旧方法の選択にあたっては，すべてのシステム構成要素を初期化する組合せの中で，最短で実施できるものを採用する．
- 選択した方法の実施順は，運用ポリシに従う．

以上から，障害復旧フローの作成を復旧方法の選択とその実施順の決定に分けて説明する．

4.1 復旧方法の選択

復旧方法の選択は，次のように定式化することができる．いま，復旧方法を j（$1 \leq j \leq M$）とし，その実施に要する時間を $T(j)$ とすると，任意の障害復旧フローを最後まで実施するのに要する累積時間は次式で与えられる．

$$\sum T(j) \times x_j \tag{1}$$

このとき，x_j は 0 もしくは 1 のいずれかをとる変数であり，障害復旧フローに含まれる復旧方法の場合が 1，そうでない場合は 0 とする．

提案方法では，障害発生時に原因となっているシステム構成要素は未特定の前提であるので，障害復旧フローはシステム構成要素全体を初期化するように復旧方法を選択する必要がある．すなわち，システムの構成要素の集合を V とすると，求める復旧方法の組合せは，システム構成要素全体を初期化することのできる組合せのうち，式 (1) を最小とする $\{j\}$ となる．

復旧方法 j が初期化するシステム構成要素の集合を R_j とすると，システム構成要素全体を初期化する制約条件は次式で表すことができる．

$$R_1x_1 \cup R_2x_2 \cup \cdots \cup R_Mx_M = V \tag{2}$$

ただし，

$$R_jx_j = \begin{cases} R_j & (x_j = 1) \\ \{\} \text{ 空集合} & (x_j = 0) \end{cases}$$

とする．

以上は，式 (2) を満たしつつ（制約条件），式 (1) を最小とする（目的関数），組合せ最適化問題であるので，動的計画法（Dynamic Programming）を用いて解くことができる．動的計画法とは，最適化問題を複数の部分問題に分割して解く際に，そこまでに求められている以上の最適解が求められないような部分問題を切り捨てながら解いていく手法である [6]．本問題への適用は次のとおりである．

いま，復旧方法 1 から復旧方法 j の中から 1 つ以上の復旧方法を用いて初期化できる最大初期化範囲を Q_j，その組合せのうち，実施時間が最小となる最小累積時間を $P(j, Q_j)$ として，これらを j に対して帰納的に求め，最終的に $P(M, V)$ を得る．

まず，$j = 1$ の場合について述べる．この場合は復旧方法 1 を施すのみであるので，$P(1, Q_1)$ と Q_1 は次の枠内のようになる．

> $j = 1$ の場合
>
> $P(1, Q_1) = T(1) \quad Q_1 = R_1$

次に，復旧方法 j が 2 以上の場合について述べる．この場合，復旧方法 j を実施することによって，初期化するシステム構成要素が増加する場合とそうでない場合があり，

それにより求める方法が異なるため，分けて述べる．

初期化するシステム構成要素が増加する場合，復旧方法 j を実施することで初期化範囲は $Q_{j-1} \cup R_j$ となる．また，増加するシステム構成要素を初期化するには復旧方法 j が必須であるので，最小累積時間は $T(j)$ と復旧方法 j 以外（復旧方法 $1 \sim j-1$）の時間の合計となる．後者の最小値を計算するには，復旧方法 j で増加するシステム構成要素以外も初期化することを想定し，その範囲を部分的に除いた範囲を復旧方法 $1 \sim j-1$ で初期化するのに必要な最小時間を達成する組合せを求めればよい．したがって，$P(j, Q_j)$ と Q_j は次の枠内のようになる．なお，この場合のシステム構成要素と復旧方法 $j-1$ までの初期化範囲，復旧方法 j の初期化範囲，再計算する範囲の関係を**図 2** (a) に示す．

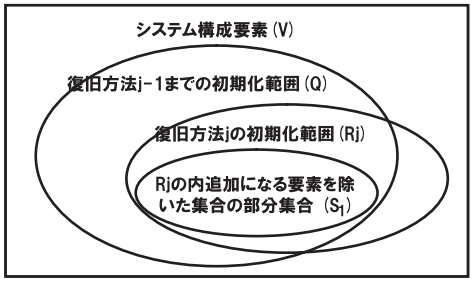
$j = 2 \sim M$ で，初期化範囲が増加する場合

$$P(j, Q_j) = \min\{ T(j) + P(j-1, Q_{j-1} - S_1) \quad //j \text{ の初期化範囲を除いた範囲で再計算} \}$$

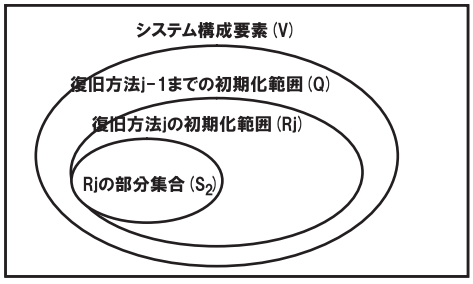
$$Q_j = Q_{j-1} \cup R_j$$

ただし，S_1 は R_j のうち追加になる要素を除いた集合の部分集合．

初期化するシステム構成要素が増加しない場合は，復旧方法 j を加えることによって，それまでの復旧方法の組合せよりも短い時間で復旧できるか否かを調べればよい．すなわち，復旧方法 j と他の復旧方法（復旧方法 $1 \sim j-1$）の組合せのうち，時間が最小となる組合せを算出し，それまでの最小累積時間（$P(j-1, Q_{j-1})$）と比較して小さい方を採用する．



(a) 初期化範囲が増加する場合



(b) 初期化範囲が増加しない場合

図 2 初期化範囲が増加する場合と増加しない場合

Fig. 2 Increased initialization areas and no increased initialization area.

復旧方法 j と復旧方法 $1 \sim j-1$ の組合せの中で実施時間の最小値を求めるには，復旧方法 j を必ず実施することを前提として，初期化範囲（Q_{j-1}）から復旧方法 j の初期化範囲の部分集合（S_2）を除いた範囲を，復旧方法 $1 \sim j-1$ で初期化する組合せのうち，最小の累積時間となる組合せを求めればよい．

以上の議論から，初期化するシステム構成要素が増加しない場合の $P(j, Q_j)$ と Q_j は次の枠内のようになる．なお，この場合のシステム構成要素と復旧方法 $j-1$ までの初期化範囲，復旧方法 j の初期化範囲，再計算する範囲の関係を図 2 (b) に示す．

$j = 2 \sim M$ で，初期化範囲が増加しない場合：

$$P(j, Q_j) = \min\{ P(j-1, Q_{j-1}) \quad //\text{前回までの計算値}, \; P(j-1, Q_{j-1} - S_2) + T(j) \quad //j \text{ の初期化範囲を除いた範囲で再計算} \}$$

$$Q_j = Q_{j-1}$$

ただし，S_2 は R_j の部分集合．

4.2 復旧方法の実施順の決定と運用ポリシの仮定

前節で選択した復旧方法の実施順を決定するにあたっては，3 章 (3) で述べたとおり運用ポリシに従う．

一般に，運用ポリシはシステムごとに異なることが想定され，共通的な項目を前提とすることは困難である．しかしながら，本論文で対象としているシステム（図 1）を前提とし，障害復旧に限定することで，提案する方法に必要な項目を仮定することができる．そこで，以降の議論を進めるために，汎用性を損なわない範囲で，運用ポリシの一部を仮定することとする．

図 1 のシステムの主目的はユーザからの利用要求に対して正常に応答するサービスを継続的に行うことである．これに基づいて次の 3 つの運用ポリシを仮定する．

① サービス提供に支障をきたしている時間（り障時間）を最小限とする．

② サービス影響などの理由で，復旧方法の実施順序に制約が生じることがある．たとえば，ある復旧方法は別の復旧方法に優先する，などである．

③ サービス影響が極端に大きい復旧方法に対しては，実施条件を個別に設ける．たとえば，「必ず最後に実施する」，あるいは「1 回のみ実施する」などである．

4.3 障害復旧フローの作成手順

前述までの議論をもとに，障害復旧フローを作成する手順をまとめる．

(1) 復旧方法の洗い出し

表 1 を参考に対象システムで利用できる復旧方法を洗い出す．加えて，復旧方法ごとに，サービス影響，復旧方法に要する時間，初期化範囲もあわせて洗い出す．

(2) 復旧方法の選択

前項で洗い出した復旧方法を順に追加しながら，4.1 節で示した方法に従って初期化範囲と最小累積時間を求めていき，障害復旧フローに採用する復旧方法の組合せを求める．なお，運用ポリシにより，組み合わせることができない復旧方法が含まれている場合は，その復旧方法のいずれかを外し，再選択する．

(3) 実施順の決定

前項で採用した復旧方法に対して，4.2 節で仮定した運用ポリシの ② および ③ から実施順を決定する．

5. 適用例

本論文で提案した障害復旧フローの作成方法を商用システムに適用した結果を述べる．

5.1 適用システム概要

適用したシステムは，P2P 通信を仲介する．具体的には，あらかじめ登録したユーザから別のユーザへの接続要求を受け付け，接続先の状態確認，接続，切断を司る．また，このシステムはユーザを分散して収容することができ，1 システムで最大 N ユーザを収容することができる．

このシステムは 1 台のサーバで構成され，その中でハードが二重化，ACT-SBY 構成となっている．ソフトウェアは OS，ミドルウェア，アプリの 3 階層になっている．そのほか，システムの構成要素には，4 種類のデータがある．

このシステムで使用できる復旧方法は 14 種類あり，そのサービス影響，復旧に要する時間，初期化範囲を整理したものを**表 2** に示す．サービス影響の列には，通信中の状態を維持したまま復旧できる場合を「無」，切断する場合を「有」と記述した．復旧時間は，事前準備・復旧・事後作業の時間を含む．復旧方法として，H/W 切替え，プロセスの再起動，データの初期化があり，復旧方法ごとに初期化するシステム構成要素の欄に●を記した．

5.2 適用システムの運用ポリシ

適用したシステムの運用ポリシを以下に示す．

(1) サービス影響のない復旧方法を優先

障害発生時に必ずしもすべてのサービス提供が不可能になっているとは限らず，一部のユーザはサービスを享受している場合もある．当該システムは通信の開始から終了までを管理するので，ユーザごとの状態管理を行っている．再起動などを行うと，サービスを中断し，被害を拡大させることから，サービス影響のない復旧方法を優先して行う．表 2 では，2 列目のサービス影響が「無」の復旧方法 1，2，6，10 が優先の対象となる．

(2) バックアップファイルを用いたサービス回復は最終手段

当該システムでは，サービスを利用するユーザを登録する．その登録は随時，追加・変更・削除することができる．バックアップファイルを用いて立ち上げ直すと，ユーザの登録状態はバックアップ取得時に戻る．もちろん，これを修復することは可能であるが，他の復旧方法に比べて多大な時間を要する．そこで，当該復旧方法は最終手段とし，

表 2 復旧方法一覧
Table 2 Recovery method list.

復旧方法	サービス影響	復旧時間	復旧方法							
			系切替	プロセス再起動			データの初期化			
				ｱﾌﾟﾘ	ﾐﾄﾞﾙ	OS	ﾃﾞｰﾀ 1	ﾃﾞｰﾀ 2	ﾃﾞｰﾀ 3	ﾃﾞｰﾀ 4
1	無	20	●							
2	無	500	●	●						
3	有	480					●	●		
4	有	500	●	●			●	●		
5	有	540		●			●	●		
6	無	560	●	●	●					
7	有	540		●			●			
8	有	560	●	●	●		●			
9	有	600		●	●		●	●		
10	無	740	●	●	●	●				
11	有	720		●	●		●	●		
12	有	740	●	●	●	●	●	●		
13	有	780		●	●	●	●	●	●	
14	有	1,020	●	●	●	●	●	●	●	●

表 3 復旧方法選択の経過
Table 3 Recovery method selection process.

評価する復旧方法	$P(j,Q_j)$	$T(j)$	初期化範囲が増える場合	初期化範囲に変更が無い場合	初期化範囲(Q_j)								復旧方法候補
			$P(j-1, Q_{j-1}-S_1)$	$P(j-1, Q_{j-1}-S_2)$	H/W	アプリ	ミドル	OS	データ1	データ2	データ3	データ4	
1	20	20	−	−	●								1
2	500	500	20	−	●	●							2
3	980	480	500	−	●	●			●	●			2,3
4	500	500	−	0	●	●			●	●			4
5	500	540	−	20	●	●			●	●			4
6	1,040	560	480	−	●	●	●		●	●			3,6
7	1,040	540	−	1,040	●	●	●		●	●			3,6
8	1,040	560	−	1,040	●	●	●		●	●			3,6
9	620	600	−	20	●	●	●		●	●			1,9
10	1,220	740	480	−	●	●	●	●	●	●			3,10
11	1,220	720	−	740	●	●	●	●	●	●			3,10
12	740	740	−	740	●	●	●	●	●	●			12
13	800	780	20	−	●	●	●	●	●	●	●		1,13
14	1,020	1,020	−	1,020	●	●	●	●	●	●	●	●	14

他の復旧方法を講じても回復しない場合のみ用いることとする．表 2 では，復旧方法 14 が該当する．

(3) 系切替え後は，再度系切替えしない

系切替えをともなう復旧方法は複数ある（復旧方法 1，2，4，6，8，10，12）[*2]が，それらを組み合わせると，障害が検出された系に戻ることがある．障害が発生した系に戻すことは，サービス回復に支障をきたす恐れがあるため，系切替えを偶数回実施しないこととする．

5.3 適用結果

5.3.1 障害復旧フローの算出

復旧方法の選択の経過を**表 3** に示す．表 3 は復旧方法を 1 から順に 4.1 節の定義に従い計算したものである．復旧方法 j の行では，復旧方法 1〜j までの計算結果を表している．表 3 には計算結果として，$P(j,Q_j)$ と $T(j)$ に加えて，初期化範囲が増加する場合は $P(j-1,Q_{j-1}-S_1)$ を，初期化範囲が増加しない場合は $P(j-1,Q_{j-1}-S_2)$ を記載した．また，復旧方法 j まで評価した際の初期化範囲（Q_j）とそのときに採用された復旧方法も候補として記した．

具体的な計算方法は次のとおりである．復旧方法 1 においては，$j=1$ の場合であることから $T(1)$ の値が $P(1,Q_1)$ となる．

次に $j=2$ の場合を説明する．復旧方法 2 では，表 2 から初期化範囲が増加することから，$P(2,Q_2)$ は $T(2)$ と $P(1,Q_1-S_1)$ の合計となる．S_1 は，{H/W} と {} の 2 種類であり，$P(1,Q_1-S_1)$ はそれぞれ 0 と 20 となるので，$P(2,Q_2)$ は $T(2)$ の 500 となる．

次に，初期化範囲が増加しない，復旧方法 4 の場合を説明する．この場合，前回までの候補による時間と，復旧方法 4 を用いることを前提に最小時間の復旧方法の組合せによる時間を比較し，短い方を採用する．前回までの候補は $P(3,Q_3)=980$ で，復旧方法 4（時間 500）は，復旧方法 3 までの初期化範囲全体をその復旧方法のみで初期化できることから，$P(j,Q_j)$ は 500 となる．

以下，同様に計算し，表 3 となる．

次に，運用ポリシを適用して，障害復旧フローに採用する復旧方法と実施順を決定する．表 3 から最小累積時間は 1,020 でその際の復旧方法は 14 であるが，5.2 節 (2) から他の復旧方法に先んじて当該復旧方法を実施する案は採用できない．そこで，復旧方法 14 を除くと，復旧方法 1 と 13 の組合せが最小累積時間（800）となり，運用ポリシに違反しないので，この 2 つの復旧方法を採用する．実施順においては，5.2 節 (1) を考慮して，1 → 13 とする．最終的に障害復旧フローは，復旧番号 1 → 13 → 14 となる．

5.3.2 求めた障害復旧フローの評価

得られた障害復旧フローを当該システムの運用者と議論し，その評価を行った．

得られたフローは，冗長系への切替え → システム全体の再起動 → バックアップを用いた再起動となっており，システムの運用中にサービス影響のある障害発生に対して，さしあたりサービス回復を図るうえで，経験豊富な運用者の意見と同じであり，違和感なく受け入れられた．

運用者からはサービス影響のない復旧方法をすべて最初に行う（H/W 切替え → アプリ再起動 → ミドルウェア再起動 → OS 再起動 →・・・）案も提示された．この案は，障害の回復する機会が求めた案より 1 回多く，障害原因がアプリにあった場合には，求めた案よりも早く回復するという長所がある．しかし，累積時間が 1,820 + α も要するため，採用した組合せに勝るものではないと判断された．

また，復旧方法 1 → 12 → 14 の案も運用者から提示された．この案と求めた案の両者とも 3 種類の復旧方法を実施

[*2] 復旧方法 14 も H/W を切り替えるが，再立ち上げを行い，H/W を初期化することから外した．

し，最終の初期化範囲も同じであるが，当該案の方が時間で 20 短いことが長所である．ただし，当該案は 2 回目の復旧方法の初期化範囲が求めた案に比べて小さいことと，系切替えを偶数回行い，障害系に戻ることが欠点である．運用ポリシに反することから，求めた案が有利であると判断された．

さらに，復旧番号 1 → 14 の案も運用者と議論になったが，復旧方法 14 は最終手段であり，その依存割合が高くなることへの懸念があり，採用に至らなかった．

以上から，求めた障害復旧フローが有効であると判断された．なお，今回得られた障害復旧フローは，当該システムの運用部門で実際に採用された．現在，幸いにも当該障害復旧フローを使用するに至ったことはないが，これを前提に運用の体制を構築し，実施訓練なども行っており，活用されている．

6. おわりに

本論文では，高い信頼性を要求されるインフラに使用されるコンピュータシステムの運用に際して，直面している障害から迅速にサービス回復を図るための障害復旧フローの必要性を説明し，当該システムが具備している機能を駆使しつつ，運用ポリシをふまえた障害復旧フローの作成方法を提案した．障害が発生しない，もしくはシステムの構成要素に障害が発生してもシステムのサービス提供に支障をきたさないアーキテクチャなどの研究はすでにある [1] が，本研究では運用時を対象に障害への対処方法を議論した．

また，この方法を提案するにあたっては，コンピュータアーキテクチャの知識と動的計画法の技術を前提とした．従来，コンピュータシステムの運用にあたっては，経験豊富な保守者に依存することが多かったが，提案した手法のように，ソフトウェア工学の知見を利用することによって，そのスキルの一部を定式化することができた．

今後は，当該方法を他のシステムにも適用していくとともに，コンピュータシステムの保守運用にソフトウェア工学の知見を利用する課題に挑戦していく予定である．

謝辞 提案した手法を商用システムに適用した結果の評価に協力いただいた NTT 東日本広域ネットワークセンタサービスマネジメント部門の諸氏に感謝します．

参考文献

[1] 南谷 崇，フォールトトレラントコンピュータ，オーム社 (1991).
[2] Shooman, M.L.: *Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design*, Wiley (2002).
[3] 松田 実，落合民哉：交換機における二重化冗長構成の検討，電子情報通信学会秋季大会講演論文集 1994 年，通信（2），p.17 (1994).
[4] 敷田幹文，井口 寧，三輪信介，丹 康雄，松澤照男：大規模高可用性サーバの設計と運用，情報処理学会分散システム/インターネット運用技術シンポジウム 2001，pp.57–62 (2001).
[5] IPA：重要インフラ情報システムの信頼性向上の取り組みガイドブック，入手先 ⟨http://sec.ipa.go.jp/reports/20110330.html⟩.
[6] Bellman, R.: The theory of dynamic programming, *Bull. Amer. Math. Soc.*, Vol.60, pp.503–516 (1954).
[7] Deris, M., Rabiei, M., Noraziah, A. and Suzuri, H.M.: High service reliability for cluster server systems, *IEEE International Conference on Cluster Computing*, pp.280–287 (Dec. 2003).
[8] 山田茂樹，宮保憲治，久保田稔：交換ノード技術とネットワーク，信学誌，Vol.81, No.4, pp.336–342 (1998).
[9] 渡辺信幸，岸田好司： 公衆網に適した VoIP 通信システムの構成法，信学論（B），Vol.J86-B, No.10, pp.2126–2133 (2003).
[10] Yamane, K., Furukawa, T. and Nishizano, T.: Service management system for IP service, *NTT Technical Review*, Vol.1, No.4, pp.50–58 (2003).
[11] Oshikiri, K., Honma, Y., Oimatsu, T. and Nishizono, T.: Operation systems for type-X and the MPLS network, *NTT Technical Review*, Vol.1, No.5, pp.50–58 (2003).
[12] 今田美幸，増田悦夫，増尾和行，三木修次：強制再開移行論理に関する一考察，電子情報通信学会秋季大会講演論文集，情報・システム，151 (1994).
[13] 川原洋人，柴垣 斉，仲谷 元，大石和寛： システム高速再開における端末無中断方式，情報処理学会論文誌，Vol.30, No.2, pp.214–225 (1989).
[14] 今田美幸，増尾和行，三木修次，古川 誠： 系切り替え制御方式の検討，電子情報通信学会技術研究報告，SSE，交換システム，Vol.93, No.72, pp.43–48 (1993).
[15] 高野 誠，小野大泰，斎藤 勲，藤田勝美：分散通信システムの再開処理方式の検討，電子情報通信学会技術研究報告，SSE, 交換システム，Vol.94, No.2, pp.13–18 (1994).
[16] 中村俊郎，岡 英一：AIN における高信頼ノード構成条件に関する検討，電子情報通信学会技術研究報告，SSE，交換システム，Vol.95, No.216, pp.19–25 (1995).
[17] 木村伸宏，山田 祥，瀬社家光，西園敏弘：IP 通信サービスのための高可用化サーバプラットホーム，電子情報通信学会論文誌 B，通信，Vol.J88-B, No.1, pp.224–233 (2005).
[18] 宮山 哲，日比野裕二：交換機故障における運用データ復旧方式の検討，電子情報通信学会秋季大会講演論文集，通信（2），p.112 (1994).
[19] 岡 利幸，野中 章：加入者データの復旧方式の高度化に関する一考察，電子情報通信学会総合大会講演論文集，通信（2），p.104 (1995).
[20] 小池友岳，白鳥 毅，鈴木俊範，吉本正明，安藤智和，水上貴司，尾崎裕二：キャリアグレード・サービス・プラットフォーム，沖テクニカルレビュー 2005 年 1 月/第 201 号，Vol.72, No.1, pp.24–29 (2005).

長野 伸一 （正会員）

1988 年山梨大学大学院修士課程計算機科学専攻修了．同年 NTT 入社．NTT 東日本研究開発センタ勤務．IP 通信網の運用に関する技術支援に従事．工学博士．電子情報通信学会，プロジェクトマネジメント学会，ACM 各会員．

谷口 友宏

2000 年東京大学工学部精密機械工学科卒業．2002 年同大学大学院精密機械工学専攻修士課程修了．同年 NTT 入社．現在，NTT 東日本研究開発センタに勤務．IP 電話のシステム/サービス創出に関する研究開発に従事．工学博士．

谷 洋介

1997 年上智大学理工学部電気・電子工学科卒業．1999 年同大学大学院電気・電子工学専攻修士課程修了．同年 NTT 入社．現在，NTT 東日本研究開発センタに勤務．IP 通信網の運用をサポートするシステム開発に従事．

葛西 圭祐

1997 年北海道大学工学部機械工学科卒業．同年 NTT 入社．現在，NTT 東日本研究開発センタ勤務．IP 通信網の運用に関する技術支援に従事．

三堀 英彦

1996 年早稲田大学大学院電子通信学専攻修士課程修了．同年 NTT 入社．現在，NTT 東日本研究開発センタに勤務し，IP 電話のシステム/通信方式に関する研究開発に従事．電子情報通信学会会員．