3E-5

障害対処のためのシステム状態管理ユーザインタフェース

池上 輝哉　　加藤 清志　　中村 暢達　　平池 龍一

ＮＥＣ インターネットシステム研究所

1. はじめに

社会インフラ化が進んでいる近年の IT システムでは、たとえ障害が発生しても即座に復旧可能とする、あるいは顕在化させないといった高い耐障害性が要求される。障害を引き起こす原因には、運用管理者の判断・操作ミス等、これまで主にハードウェア領域において強化されてきた耐障害性の機能だけでは充分に対応できないものが多く含まれており、運用を支援するユーザインタフェース（UI）が重要な役割を果たすようになってきた。

従来、運用管理の現場においては、知識と経験を備えた専門家が作業にあたり、GUI よりも作業効率重視のコマンド型 UI が好まれる傾向にあった[1]。しかし、運用開始後しばらくすると、システムのハードウェア構成や使用ソフトウェアが大きく変わることもあり、運用管理者の技量のみに頼る方法では判断ミスや操作ミスを避けられない。このようなミスを防ぐために、適切な情報提示を行う GUI が重要となる。

著者らは、アプリケーションの入出力データと処理データを自動的に記録することで、運用管理者（操作者）の操作ミスやアプリケーションバグ等に起因する障害が発生した際にも、任意のシステム状態に戻ることを可能とするシステム状態管理機構を開発した。本稿では、このシステム状態管理機構を用い、まず障害が発生した際に処理データを記録時の状態に戻し、続いて以後の入出力データを分析、障害原因となった入力データを除去することで、障害が再度発生することを回避した上で、障害発生直後のシステム状態へ復旧させることを容易にする UI を提案する。

2. 障害対処のためのシステム状態管理

本章では、任意のシステム状態に戻ることを可能とするためのシステム状態管理のあり方について考察した後、開発したシステム状態管理機構について述べる。

2.1. システム状態の管理

「システム状態」を構成する要素には、アプリケーションの動作状態と、アプリケーションが記録／参照するデータとがある。

適当な周期毎にアプリケーションデータのバックアップ（チェックポイント）をとれば、その時点へのデータ復旧は保証される。しかし、エンドユーザからのリクエストを絶え間なく処理するようなシステムにおいては、単にデータを過去の状態に戻すだけでは、バックアップ以後に処理されたリクエストについてはアプリケーションデータに反映されていないことになる。従って、任意のシステム状態への復旧を可能とするためには、アプリケーションのデータだけでなく動作状態も併せて記録し、管理することが必要である（図 1）。

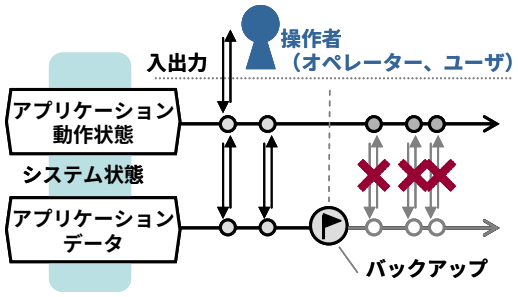

図 1　アプリケーションデータと動作状態

動作状態には、アプリケーション内部の一時的な状態もあるが、サービス継続の観点では、外部との対話でコミットされた情報は保持すべきであり、アプリケーション入出力データ(ウェブクエリや DB トランザクション)の記録が必要である。アプリケーションへの入力によって障害が引き起こされるパタンには、オペレーターの操作・設定ミスから誤作動やサービス停止に繋がるものと、エンドユーザからのリクエストなど特定の入力シーケンスによって、開発段階では発見されなかったバグが活性化するものがある。従って、入出力データとして、運用管理を行うオペレーターの操作と、サービスを享受するエンドユーザのリクエストの両方を管理すべきである。

2.2. システム状態管理機構

前述のシステム状態管理を実現するために、著者らは Web/AP/DB の三層構造モデルを対象としたシステム状態管理機構を開発した（図 2）[2]。

本機構は、アプリケーション入出力インタフェースのプロトコルとして一般的に使われている HTTP を対象とし、オペレーターやエンドユーザからの処理要求および応答をウェブクエリ履歴として記録し、後から再生（クエリ再実行）することを可能とする。また、アプリケーションデータを、仮想マシンのスナップショットという形態で、チェックポイントとして記録する。チェックポイントを記録する際に、最後に処理したクエリの情報を併記することで、チェックポイント読み出し時に、再生を開始するクエリを特定し、所望の状態に到達するまで再生を繰り返すことで、任意のシステム状態へ戻す。

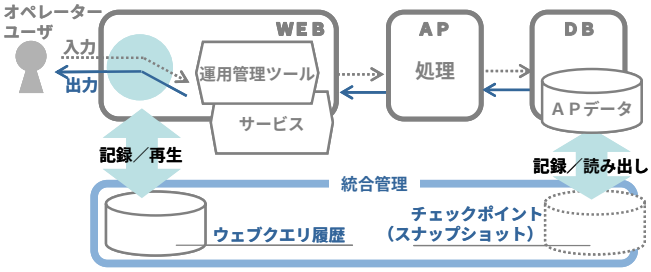

図 2　システム状態管理機構

A System Management User Interface for Service Failure Recovery
Teruya IKEGAMI, Kiyoshi KATO, Nobutatsu NAKAMURA and Ryuichi HIRAIKE
Internet Systems Research Laboratories, NEC Corporation.

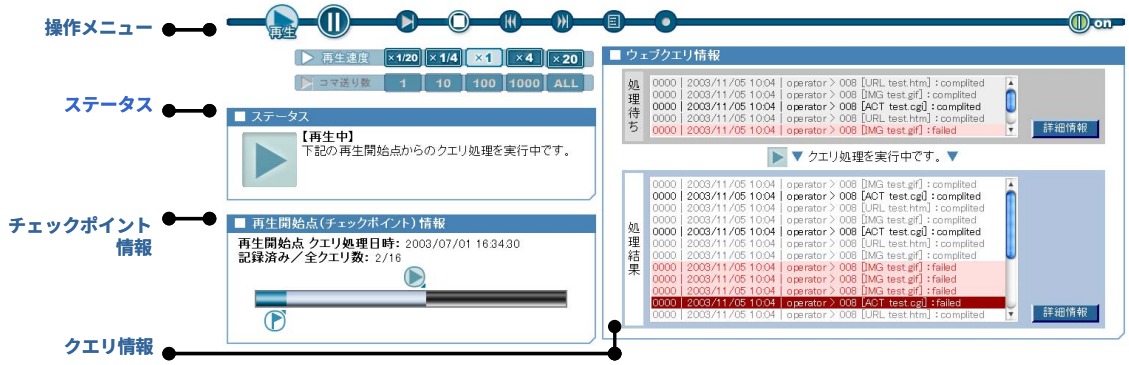

図 3 基本画面構成

3. システム状態管理ユーザインタフェース

本章では、システム状態管理機構上に開発した、任意のシステム状態への復旧操作を可能とする UI について述べる。

3.1. 表示画面と操作方法

図 3 に本 UI の基本画面を示す。システム状態管理機構を制御するために必要十分な情報として、クエリやチェックポイントに関する情報を基本画面に配している。

本 UI では、「再生」だけでなく、クエリ処理制御やチェックポイント管理の操作に「コマ送り」や「巻き戻し」、「記録」といった音楽・映像編集で用いられるメタファを対応付けた操作メニューボタンを設けた。これにより、クエリ実行（再生）、リストア（巻き戻し）、チェックポイント保存（記録）等の操作をワンボタンで実行可能とした。また、クエリ処理が停止状態にある場合において、個々のクエリの詳細情報を参照することや、再生時に処理するか否かを設定することを可能とした。

このような柔軟なクエリ処理制御と、アプリケーションデータの記録・読み出しを可能とするチェックポイント管理とを併せて備えることにより、障害発生時にチェックポイントまでシステム状態を戻すだけでなく、障害の原因となったクエリの発見や除去の支援が容易になる。

3.2 障害対処時の操作例

オペレーターの操作ミスや、ある種の入力シーケンスによるアプリケーションバグの活性化により、システムに問題が生じた場面を想定し、本 UI を用いて障害対処を行う際の操作の流れを示す（図 4）。

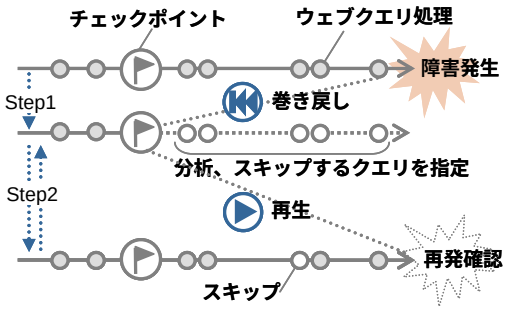

図 4 障害対処時の操作例

ＳＴＥＰ 1：チェックポイントの読み出し

障害対処を行う操作者は、障害発生前の正常に動作していた時点で記録したチェックポイントを読み出す（巻き戻し）。しかし、単にチェックポイントを読み出しただけでは、チェックポイント以後のクエリの分だけ、障害発生時点のシステム状態との間に差分が存在することになる。そこで、チェックポイント以後のクエリを処理（再生）することで、障害発生直後のシステム状態に戻すことが可能となるが、当然ながら先に発生した障害が再度発生することとなってしまう。

ＳＴＥＰ 2：障害原因の分析

クエリ再生時に障害が発生しないようにするためには、障害を引き起こしたクエリを取り除く必要がある。まず、チェックポイント以後の個々のクエリを参照し、障害の引き金となった可能性のあるクエリを洗い出す。次に、疑わしいクエリを再生時にスキップするように設定した上で再生し、同様の障害が発生するか否かを確認する。障害が発生した場合、スキップしなかったクエリに原因があることになるため、再度チェックポイントまで復帰して同様の作業を継続する。

障害の原因となったクエリを特定できれば、このクエリを再生時にスキップすることで、障害が再度発生することを回避した上で、障害発生直後のシステム状態へ復旧させることが可能となる。

4. おわりに

本稿では、アプリケーションの動作状態とデータとをあわせて記録することで、たとえ障害が発生しても過去のシステム状態に戻り、障害が再度発生することを回避した上で、障害発生直後のシステム状態へ復旧させることを容易にするシステム状態管理 UI を提案した。

今後、本 UI について、操作の間違え難さや学習容易性、操作効率などを評価するため、実際の運用管理現場での試用を予定している。

参考文献

[1] Rob Barrett, Paul Maglio, Eser Kandogan, "Usable Autonomic Computing Systems: the Administrator's Perspective", Proceeding of ICAC'04, 2004

[2] 中村,加藤,平池、"自律運用管理におけるサービス指向多重実行方式"、並列/分散/協調処理に関するサマーワークショップ（SWoPP)、2004