自律運用管理に向けたポリシー適用優先度の制御に関する一考察

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

情報通信サービスの社会インフラ化が進むにつれて、連続運用可能な高信頼システムが望まれているが、大規模かつ複雑に連携したシステムでは、従来のような入念なテストと２４時間監視体制といった運用方法が困難になりつつある。このような状況を受けて、自律管理システムに関する様々な研究が進められており、サーバや運用管理ソフトにも一部機能が搭載されつつある[1]。

筆者らは、次世代の自律運用管理システム実現に向けた研究を進めており、オペレータが入力したルールに従って個々の機器設定を自動的に最適化するためのポリシー優先度制御方式を試作した。本論文では、自律管理の課題と試作システムについて述べる。

２．自律管理の課題

大規模かつ相互連携したシステムでの手作業運用は困難になりつつあり、現状のシステム停止原因の大部分が操作や設定のミスによるものであるとの調査報告もある。このような複雑なシステムの運用支援には、従来のような個々の機器単位での自動化だけでは十分ではなく、システム全体としての新しい枠組みの確立が必要となる。

2.1 自律ループの実現

自律管理では、人手に委ねられている運用作業を代行する自律ループ（状態検出→判断→対処実行の流れ）の実現が必要となる。知的な判断についてはＡＩ手法として様々な研究がなされているが、実システムの状態を適切に把握し、対処を実システム上で実行するといった前処理／後処理の部分との整合性を考慮して新たに検討する必要がある。

2.2 障害対処時の対話

一般的に、処理の自動化では、状況判断のすぐ後に対処実行となるが、実際の作業では、対処実行までにいくつかのステップを踏むことが多い。高信頼性を要求されるシステムでは、状況判断の後に障害システムから正常システムへの業務切換が行われる。その後、障害システムの根本原因の追究と再現性検証、復旧計画立案と顧客承認を経て対処が実行される。これらのステップでは人（管理者）との対話が密に行われており、障害対処を自動化する場合も、管理者との対話を考慮する必要がある。適切な状況提示や対話があれば、管理者が処理をシステムに委ねて良いかどうかを判断し易くなり、より円滑に自動化が実現できる。

2.3 操作ミスへの対応

現状のシステムではミスに対する防御機能が十分とは言えない面がある。障害が操作ミスに起因している場合、管理者の責任として手順見直し等で対応が行われている。しかし、経験を積んだ管理者でもミスは不可避なものであり、適切な情報提示や対話手法によってミスを低減する機構や、操作ミスが起こることを前提としたUNDO（後戻り）機構が必要である。

2.4 ソフトウェア修正の支援

サービス運用中のシステム修正、特に、ソフトウェア・バグの修正が管理者の負担となっている。ハードウェア障害の場合は、故障部品を単体検査後の新しい部品と置き換えることで復旧できるが、ソフトウェアは業務内容と関係が深いため、一部分を変更するとサービスの挙動が変わる場合がある。原因を特定できたとしても、修正して問題ないかどうかを判断することが難しく、復旧の遅れや累積障害の引き金となっている。対策としては、元の状態に戻す従来型の復旧処理ではなく、運用時の新機能テストや旧機能への復帰円滑化といった新環境への移行支援が必要である。

A Study of the Priority Optimization for Policy-based Self-managing System
Kiyoshi KATO, Nobutatsu NAKAMURA and Ryuichi HIRAIKE
Internet Systems Research Laboratories, NEC Corporation.

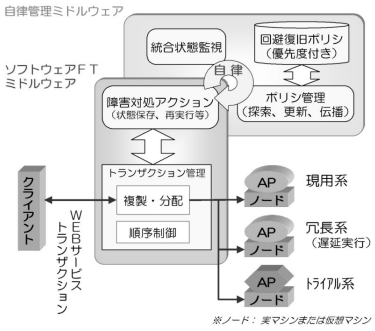

図１　試作システムの構成

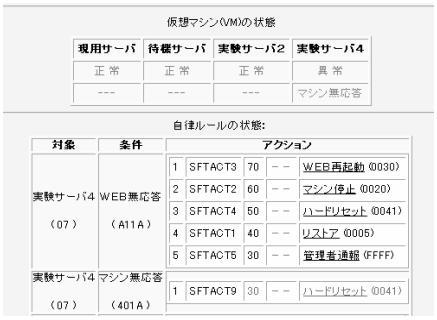

図２　ルール表示画面例

３．自律管理フレームワーク

前述した課題を考慮した基本構成として、

①ルール優先度制御を用いた自律管理機能

(→ 自律ループと対話機能を提供)

②AP 多重実行によるサービス FT 機能[2]

(→ 操作ミス対応とソフト修正支援)

を持つ実験システムを試作した（図１）。これらの機能はミドルウェアとして実現し、サービスの無応答といった障害状態を検出し、プログラムの起動やマシンの再起動／リストアといった対処を自動的に実行する。以下では、①の自律管理の機能について述べる。

3.1 ポリシー優先度制御方式

対処の決定は、Condition-Action 型ポリシー（if-then 形式の障害対処ルール群）に従って自律管理ミドルウェアが行う。障害対処ルールで、同じ障害(Condition)に対して優先度の異なる複数の対処候補(Action)を用意することで、複数候補を選択する権限をシステムに付与することができる。対処コマンド実行後は、状態を再検知して異常が回復したかどうかを判定し、その結果に応じて各ルールの優先度を変更する。優先度制御に実際の状態検出を用いることで、自ら決定した対処に対するフィードバック機構となる。これにより、障害対処の自動化とともに、実際の運用環境に応じたルール修正も自動化することができる。

3.2 管理者との対話

システムが自律的に対処している状態は、ルール一覧画面（図２）の優先度変化として管理者に伝えることができる。これは、障害発生時にどう対処しているかという過程に加えて、どのルールが有効であるかを一定期間検証した結果でもある。管理者は、入力したルールが現在のシステムで有効かどうかを把握した上で、必要に応じてルールを修正することができる。また、本システムでは、システム自身が対処成功を確認できなければ、優先度が低下して、次回以降の対処時に当該ルールが選択されない状態となる。このため、管理者が間違ったルールを入力した場合でも、システムの信頼性を極端に低下させることがない。ルールの厳密な検証は管理者の大きな負担であるが、本システムでは、信頼性を損なうことなくこのよう負担を低減することができる。

3.3 考察

今回の試作では、リアルタイム性を考慮して優先度情報を単一の数値として実現したが、対処効果による優先度制御を継続することで、他のルールと比べて安定した効果が得られるルールを選択できた。発生頻度の多い障害のルールが上位に来ることで平均的な対処時間が短縮されるとともに、稀な障害に対しても、順次下位のルールを試すことで対処可能であった。また、成功する可能性の高いルールが上位となることで、間違った対処による障害の波及を防ぐ効果も得られた。

４．おわりに

大規模化するシステムの運用管理コスト低減を目的として、運用ポリシーによる管理自動化と管理者の負担低減について考察した。対処効果に基づくポリシー優先度制御を実現した試作システムは、処理の自動化による信頼性向上に加えて、管理者のノウハウをルールとして収集する上での効果も期待できる。今後、ポリシー抽象度制御や伝播といった機能の開発を進める。

参考文献

[1] Studwell, T., “Orchestrating Self-Managing Systems for Autonomic Computing: The Role of Standards”, 14th IFIP/IEEE DSOM2003, Oct.2003.

[2] 中村他,” 耐障害に向けたサービスアプリケーション多重実行方式の提案”, 情処 66 全大, 5D-1, Mar. 2004.