O-019

監視システムにおける障害箇所特定
Fault Item Identification in IT System monitoring

山田　耕一†　　川岸 諒子†
Kouichi Yamada　　Ryoko Kawagishi

1. 背景

データセンターでは、顧客が構築したシステムの監視サービスを提供している場合がある。監視対象となるネットワークデバイスまたはサーバを監視装置で監視し、そこから通知されるアラームに基づいて顧客への通知を行ったり、障害対応を行ったりする。このような IT サービスでは、ITIL(Information Technology Infrastructure Library) [1] [2] [3] [4] [5]や ISO20000(ITSMS：Information Technology Service Management Systems)等の標準を導入するケースが多い。このような標準を使用して、運用監視サービスの標準化、自動化を行う場合、構成管理データベースを整備して、イベント処理の対応などを行うと効率がよい。

2．従来の監視での課題

2.1　データセンターでの監視業務

データセンターにおける監視業務では、監視対象となるネットワークデバイスまたはサーバを監視装置で監視し、そこから通知されるアラームに基づいて顧客への通知、障害対応等を行っている。

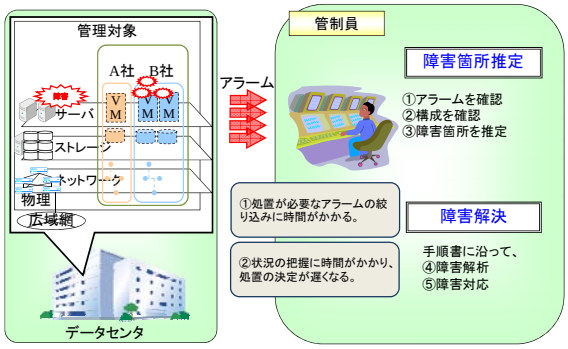

図 1. サーバ監視の例

障害対応では、アラーム発生時に、まずアラームの内容から重要度、緊急度などを確認する。対応が必要であると判断されたら、アラームが発生したサーバまたはネットワークデバイスの情報を、構成管理データベースに保存された、監視用の構成情報で確認し、どの機器で障害が発生しているかを推定する。その後、手順書に従って障害の解析、対応などを行う。監視センターで対応出来ない物については、サーバまたはネットワークデバイスを管理している部門または顧客へアラームの通知を行う。

2.2　構成情報とイベント情報からの障害箇所特定

アラーム情報と構成管理データベースのモデル情報を使って、複数発生したアラームの中から、原因箇所を特定することが出来る。特定する手法はいくつかあるが、ここでは構成情報から距離行列を求めて、障害箇所を特定する方式について述べる。

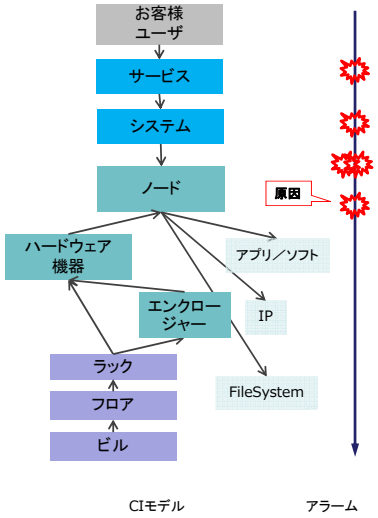

図 2. 障害箇所特定機能の概要

構成情報のモデルは、CI(Configuration Item：構成要素)と、CI 間の関係を示す依存関係から出来ている。モデルは、顧客がサービスを利用し、サービスはシステムで構成され、システムはノードで構成され...といった、システムの一般的な形態を表現する。上位の CI は下位の CI に対して機能やリソースなどに依存しているため、下位の CI の障害は、上位 CI に伝搬する。そのため、最も下位の CI で、アラームが発生している CI が障害箇所となる。通常、最上位の CI はシステムを利用する顧客を示すため、一意に定まる。そこで、最も下位の CI は、最上位の CI から最も遠い CI として求めることが出来る。最上位から最も遠い CI を求めるには、CI 間の距離を計算する必要がある。このアルゴリズムでは、構成情報から隣接行列を生成し、さらに距離行列に変換して、CI 間の距離を決定する。

距離行列が求まったら、イベント情報とのマッピングを行う。アラームを出している CI 中で、最上位の CI(たとえば、A 社)からの距離が最も大きい CI が障害箇所となる。具体的な例で説明する。

図 3 に、サーバ 7 台が Fire Wall に接続しているシステムの例を示す。このシステムの距離行列を表 1 に示す。ここで、Fire Wall(k)に障害が発生し、それに伴ってサーバ 1～7(d～j)についても疎通確認が出来ないというアラームが発生したとする。この場合、サーバ 1～7 と Fire Wall でアラームが発生した状態となる。このアラーム情報を距離行列

†三菱電機（株）情報技術総合研究所

にマッピングする。なお、この表 1 では、表記の都合上 CI を図 3 中に示した a～k の記号で表している。

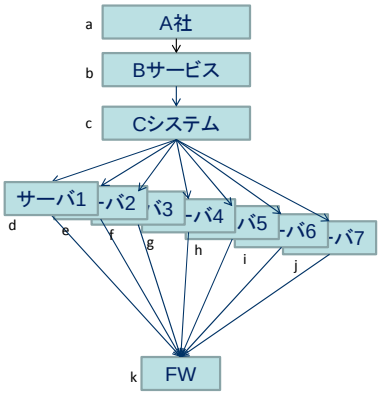

図 3. システム構成の例

表 1. アラームを距離行列にマッピング

	a	b	c	d	e	f	g	h	i	j	k
a											
b	1										
c	2	1									
d	**3**	2	1								
e	**3**	2	1	0							
f	**3**	2	1	0	0						
g	**3**	2	1	0	0	0					
h	**3**	2	1	0	0	0	0				
i	**3**	2	1	0	0	0	0	0			
j	**3**	2	1	0	0	0	0	0	0		
k	**4**	3	2	1	1	1	1	1	1	1	

表中、最上位の CI である「A 社」を示す「a」から最も遠くでアラームが発生しているのは「k」であり、元の構成情報中では Fire Wall が障害箇所である事が判断出来た。この方式では、一旦距離行列を求めたら、システムの構成変更があるまで再計算は必要なく、アラームとのマッピングのみで障害箇所特定が可能である。

2.3　障害箇所特定の課題

2.3.1. アルゴリズム上の課題

障害箇所特定のアルゴリズムが正しく動作するには、以下の条件が前提となる。

- システムを構成するすべての CI が、構成管理データベースに登録されている
- CI 間のすべての依存関係が、構成管理データベースに登録されている

また、障害箇所特定のアルゴリズムは以下の制約事項がある。

- 監視対象外の CI が障害の原因箇所である場合、正しい障害箇所特定が出来ない
- 複数の障害が同時発生した場合、正しい障害箇所特定が出来ない

なお、他の今回紹介したアルゴリズム以外の場合でも、システムの正確な構成情報が無いと、障害箇所特定が出来ない事は変わらない。

2.3.2. 監視サービス上の課題

自社システムの監視などの場合は、個々のサーバまたはネットワークデバイスの情報について詳細な構成情報を得る事が出来る。

しかし、ハウジングサービスにより顧客資産のシステムをデータセンターに設置している場合や、仮想サーバを貸しているだけで、それぞれのサーバの利用方法には関わっていない場合等では、各サーバやネットワーク機器の詳しい情報や、お互いがどのように依存しているかについての情報を、顧客から得られない事がある。

そのため、前項のアルゴリズム上の課題により、以下に挙げる様なサービス運用上の課題が発生する。

ネットワーク障害

- ネットワークの障害がどのサーバ、アプリ、サービスに影響するのかが把握できない
- レイヤ 4 （トランスポート層）までの障害把握に留まり、アプリケーション間の通信障害は把握できない

サーバ障害

- サーバの影響がシステムやサービスへどのように影響するのかを把握できない
- 発生した障害がサーバ自身によるものかネットワークによるものなのかが分からない

アプリケーション障害

- アプリケーションの異常が、アプリ自体かサーバ／ネットワーク等のインフラが原因なのかを把握できない

ファシリティの障害

- 電源の故障や停電時のシステム・サービス影響度が把握できない
- 遊休資産の把握が難しく、利活用が促進されない

多数障害発生時

- 単一の根本原因となる障害と、その影響による副次的な障害が発生し、多数のアラームが通知されると、処置が必要となるアラームの絞り込みに時間がかかる。また、全体の状況把握にも時間がかかり、処置の決定が遅くなる。

3. ルールによる構成情報補完方式の検討

2 章で述べた様に、アルゴリズム上の課題と監視サービス上の課題があるため、すべてのシステム、障害ケースにおいて障害箇所特定機能が有効に働くわけではない。

そこで、不完全な構成情報に対し、ルールによって各 CI の持つ役割と、依存関係を追加し、構成情報を補完することで、障害箇所の特定を可能とする方式を検討した。この方式では、障害箇所を特定する時に、構成管理データベースの外側でルールを適用し、構成情報を変形させ、距離情

報をリストで管理することで、構成管理データベースのモデルを変更することなく障害箇所の推定が可能となる。

3.1. ルールによる構成情報の変形

CI に対して属性を追加し、ルールによって構成情報の変形を行う。変形した構成情報は、元の構成管理データベースには反映せず、アラーム情報とのマッピング用として使用する。

各 CI がどのような役割を持つかの情報を、ルールによって追加属性に持たせる。属性の追加例を以下に示す。

属性追加ルール例
- サーバ 1 は DB サーバである
- サーバ 2 は Web サーバである

次に、追加された属性を使い、構成アイテムの追加や、CI 間の依存関係追加などを行う。ルールの例を以下に示す。

構成情報変形ルール例
1. ノードが DB サーバであれば、新たにアプリ(DB)の CI を追加。アプリ→ノードの依存関係を持たせる。
2. ノードが Web サーバであれば、新たにアプリ(Web)の CI を追加。アプリ → ノードの依存関係を持たせる。
3. アプリ(Web)とアプリ(DB)の CI があれば、アプリ(Web) → アプリ(Web)の依存関係を持たせる。
4. ルールによって追加されたアプリ CI に対しては、システム → アプリの依存関係を追加する。

このようなルールで図 3 の構成情報を変形させると、図 4 の様になる。

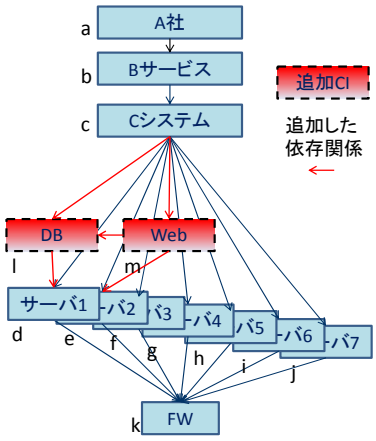

図 4. ルールによって構成情報を変形した例

ルール 1 により、DB の CI、ルール 2 により、Web の CI が新たに追加される。次に、ルール 3 により、Web→DB の依存関係が追加される。ルール 4 により、C システム→DB→サーバ 1 と、C システム→Web→サーバ 2 の依存関係が追加される。

3.2. リストによる障害箇所の特定

ルールによって変形させた構成情報を元に、障害箇所を特定する。このとき、2 章と同じ方式で距離行列を求めると、サーバ 1, 2 とも距離 3 となる。これは、ルールによって CI や依存関係の追加はあったが、元々の構成情報にあった C システム→サーバ 1 と C システム→サーバ 2 の依存関係はそのまま残しているので、最上位である A 社 CI からの距離は変化しないためである。

そこで、最上位から距離が 1 となる CI のリスト、距離が 2 となる CI のリストと、距離別に最長経路までの CI のリストを生成する。このリストは、幅優先探索などの既存のグラフ探索アルゴリズムにより生成出来る。一つの CI が異なるリストに重複して入っても良い。その場合は、CI へ到達するルートが複数存在し、そのルート毎に距離が異なる事を示している。図 4 に示す変形後の構成情報で、距離別の CI リストを生成すると、次の様になる。

距離 1 の CI：b
距離 2 の CI：c
距離 3 の CI：d,e,f,g,h,i,j,l,m
距離 4 の CI：k,d,e,l
距離 5 の CI：k,d
距離 6 の CI：k

リストが出来たら最長経路の物から順番に、アラーム発生 CI 情報と、距離の情報を比較する。今回アラームが発生していたのは、サーバ 1 とサーバ 2 であり、CI の記号では d と e である。最長経路である距離 6 のリストにはどちらも入っていない。次に長い距離 5 のリストには、アラーム発生 CI としては d が入っている。ここで、複数あったアラーム発生 CI から、一つの CI である d に絞り込む事が出来た。これはサーバ 1 を示す CI であり、サーバ 1 が障害箇所であると特定される。このようにして、ルールによって構成情報を変形し、距離別リストを用いてアラーム発生の原因となる CI を特定する。

この例では、DB サーバ、Web サーバの属性、CI、依存関係を追加した。この場合、追加した CI とサーバ 1,2 との依存関係追加により、追加 CI である DB と Web を経由するルートを通ると最上位から距離は 4 となる。さらに、Web→DB の依存関係が追加されることで、サーバ 1 は距離 5 のルートも持つ事になる。これは、間接的に、サーバ 2→サーバ 1 の依存関係を持つことを表現出来ている。

4. 考察

4.1. 他の方式との比較

システムから通知されるアラームを元に、障害箇所を特定する方式として、「特定のノードで特定のアラームが発生した場合、別のあるノードのあるアラームはその影響によるものである」というルールを設定し、そのルールに従って、原因と影響を判定する方式がある。図 3 のシステム構成の例で、以下の様なルールが定義されているとする。

- Fire Wall で障害が発生した場合、サーバ 1 に影響する
- Fire Wall で障害が発生した場合、サーバ 2 に影響する
…
- Fire Wall で障害が発生した場合、サーバ n に影響する

このようなルールを、障害が発生しうる箇所と、それが影響するすべての CI に対して定義することで、原因となる障害が発生した CI とその影響でアラーム発生している CI を区別することが可能となる。なお、ルールを簡略化するため、CI の種類や属性毎にルールを集約しても良いし、監視対象機器間の応答有無のパターンで判断しても良いし[6]、障害影響の尤度推定を行っても良い[7]。

この方式では、アラームを発生する可能性のあるすべての CI を正しく把握する必要がある。自動的に構成情報を収集する仕組みを使うなどで細かい情報をすべて集める事が出来れば対応可能となる。しかし、監視業務のみを請け負っている場合、システムの構成を得る様なツールを動作させることは出来ず、必要となる構成情報が得られない。

また、エラーの原因箇所を絞り込む方式や、システム設計時の情報を元に配置、インストール時の構成情報を生成する方式も先行技術として公開されている。しかし、これらも現状のシステムの正確な構成情報、または、設計時の正確な構成情報を使うことを前提としており、不完全な構成情報しか得られない環境に適用することが出来ない。

本方式は、不完全な構成情報に、判明した範囲で属性追加し、ルールにより CI と依存関係を追加するという変形を加えることで、障害箇所の推定を可能とするものとなっている。

4.2. 効果

この技術を監視システム中のインシデント管理に適用することにより、障害対応作業の中で、障害箇所を特定するための時間を短縮することが出来る(図 5)。

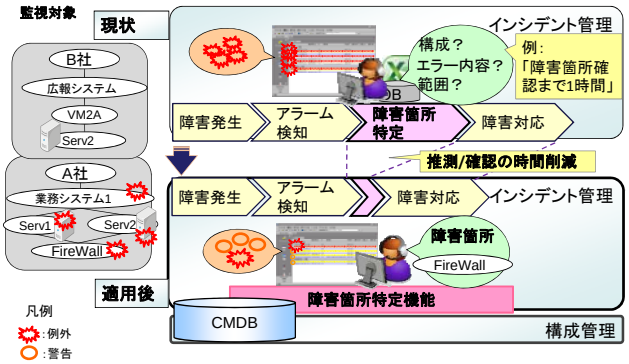

図 5. 障害箇所特定機能適用イメージ

2 章で述べたように、障害発生時には監視対象からアラーム情報が通知され、それをインシデントとして対応する。図 5 の例では、ネットワーク障害(Fire Wall)によってサーバとシステムでもアラーム発生した場合を示している。この例では、4 件アラームが発生しており、障害対応を行うため、そのうちのどれが根本となる障害箇所であるかを特定している。これは、オペレータが対象システムの構成を把握し、エラー内容から影響が及ぶ範囲を推定する、原因となる箇所を推定する等の作業を行っている。障害箇所特定機能を用いることにより、障害箇所を特定が自動的に行えるため、この時間を削減することが出来る。

2 章に挙げた課題については、以下の様に対応可能となる。

- ネットワーク障害
  現状、ネットワーク系の構成情報とサーバ系の構成情報を統一して扱い、ネットワーク障害がサーバやアプリに影響している場合でも、原因箇所を自動的に推測する事が出来る。アプリケーション情報の属性追加とルール設定により、アプリケーション間の通信障害についても把握可能となる。

- サーバ障害/アプリケーション障害
  サーバ障害とサービス障害の発生時に、サーバ障害の影響でどのサービスが影響を受けているのかを把握出来る。またアプリケーションや、ネットワーク障害が原因の場合にも切り分けが可能となる。

- ファシリティの障害
  ファシリティ関係の CI を構成管理データベースに入れる事で、電源故障や停電との関係、サービスへの影響が把握出来る。構成管理データベースの構造を変えられずファシリティ情報を参照出来ない場合でも、属性とルールを設定することで、対応可能となる。

- 多数障害発生時
  ある障害の影響により、副次的な障害が発生して多数のアラームが通知された場合、根本原因となる障害箇所が自動的に推定できるので、適切な対応によりシステムの復旧を迅速に行う事が可能となる。

5. 結論

ルールによって構成情報を補完、変形することで、アプリケーションレベル構成情報のデータが無い等、顧客資産のシステム監視で詳細な構成情報が得られない場合においても障害箇所を特定することが可能となる。これをデータセンターの監視業務に適用することで、障害対応作業において、障害箇所を特定するための時間を短縮することが出来るようになる。これにより、監視にかかるコストを下げることが可能となる。

今後は、障害箇所特定が適用出来る範囲を広げ、精度を高めて行くため、役割の属性の詳細化、依存関係追加ルールの拡充を進めていく予定である。

参考文献

[1] TSO, ITIL® 2011 edition：サービスオペレーション.
[2] TSO, ITIL® 2011 edition：サービスストラテジ.
[3] TSO, ITIL® 2011 edition：サービスデザイン.
[4] TSO, ITIL® 2011 edition：サービストランジション.
[5] TSO, ITIL® 2011 edition：継続的サービス改善.
[6] 田村智只 他, “リモート監視における障害原因箇所推定方式,” 電子情報通信学会技術研究報告, 2012.
[7] 川岸諒子 他, “監視経路の冗長性に基づく障害原因箇所推定手法の提案,” 情報科学技術フォーラム講演論文集, 2012.