大規模サーバ間の部品依存関係に基づくログ管理支援法

敷田幹文[†1] 後藤宏志[†2,*1]

近年，大企業や大学の情報システムは大規模サーバ群による集中管理を行う傾向にあり，それらのサーバにはきわめて高い信頼性が要求されている．そのような大規模システムの管理には，全体を把握する 1 人のエキスパートではなく，複数人の管理者が分担して作業する組織が増えている．各管理者は，各サーバの様々なログファイルを参照することによって，障害の解析などに必要な多くの情報を得ている．従来の統合管理ソフトウェアでは，各種サーバのログ情報を一元管理し，設定した条件に従って管理者へ通知を行うことが可能である．しかし，大規模・複雑なシステムでは膨大な量のログ情報が発生しており，その中から現在必要な情報を発見することは困難である．本論文では，サーバの各部品や様々なサービスの間の依存関係に注目し，システム全体の全ログ情報の中から管理者が現在注目している事象に関連のあるログ情報を担当区分を越えて自動抽出する方式を提案する．たとえば，Web アプリケーションの管理者がログから障害情報を発見した場合に，他の管理者が管理するデータベースサーバや大規模ストレージの膨大なログ情報の中から，今回の障害に関係のある情報のみを自動抽出して参照できるため，大規模システムにおける各管理者の障害解析作業が効率化される．また，本方式を用いた試作システムでの実験結果から，本方式を用いた大規模サーバ群のログ情報管理支援の有効性に関する議論を行う．

Management of Server Log Information Based on Dependencies among Components of Large-scale Servers

Mikifumi Shikida[†1] and Hiroshi Goto[†2,*1]

In some large organizations such as a large enterprise or a university, large-scale servers are designed to organize large information systems. Reliability of the large-scale servers is key factor of managing those systems. In general, several administrators are needed to manage those systems instead of one expert. To get needed information for an analysis of system fault, administrators have to refer many kinds of log files of servers. It often involves some servers that are managed by the others. In current software for log management, many kinds of log information of several servers are unified. Administrators will be able to get system error information by such software if they give some parameters to the software in advance. However, it will be not so easy to find real needed information because of a huge number of log information in large-scale systems. In this paper, we focus on dependencies among components of large-scale servers and services. We propose an extracting method for getting the real needed log information based on these dependencies. Our proposed method can extract specific log information in connection about details of each system failure. We illustrate our proposed method with an experimental system and discuss its usefulness.

1. はじめに

近年，情報ネットワークは我々の社会生活に必要不可欠なインフラとなっている．特に，大企業や大学の情報システムは大規模サーバ群による集中管理[1)]を行う傾向にあり，それらのサーバにはきわめて高い信頼性が要求されている[2)]．そのような大規模システムの管理には，全体を把握する 1 人のエキスパートではなく，複数人の管理者が分担して作業する組織が増えている．

ハードウェアや基本ソフトウェアの進歩によって，大規模・高信頼性サーバシステムが構築可能となったが，管理作業には人間の判断や作業が不可欠である．各管理者は，各サーバの様々なログファイルを参照することによって，障害の解析などに必要な多くの情報を得ている．特に，障害を回避する機構を持った高可用性サーバなど，複雑な構成のサーバは通常システム

†1 北陸先端科学技術大学院大学情報科学センター
Center for Information Science, Japan Advanced Institute of Science and Technology
†2 北陸先端科学技術大学院大学情報科学研究科
School of Information Science, Japan Advanced Institute of Science and Technology
*1 現在，ヤフー株式会社
Presently with Yahoo Japan Corporation

の数十倍のログ情報を出力することがあり，管理者のログ解析での負担は年々増加しているといえる[3]．

従来の統合管理ソフトウェアでは，各種サーバのログ情報を一元管理し，設定した条件に従って管理者へ通知を行うことが可能である．しかし，大規模・複雑なシステムでは膨大な量のログ情報が発生しており，その中から現在必要な情報を発見して参照することはきわめて困難である．一方，類似性や出現頻度などをもとにログ情報を加工し，視覚化を行う研究[4)–8)]がある．これらはシステム全体の状況把握時や問題の発見などにはきわめて有用であるが，個々の情報の意味に基づく処理ではないため，個別障害の解析時支援には必ずしも適していない．

本論文では，サーバの各部品や様々なサービスの間の依存関係に注目し，システム全体の全ログ情報の中から管理者が現在注目している事象に関連のあるログ情報を担当区分を越えて自動抽出する方式を提案する．たとえば，Web アプリケーションの管理者がログから障害情報を発見した場合に，他の管理者が管理するデータベースサーバや大規模ストレージの膨大なログ情報の中から，今回の障害に関係のある情報のみを自動抽出して参照できるため，大規模システムにおける各管理者の障害解析作業が効率化される．また，本方式を用いた試作システムでの実験結果から，本方式を用いた大規模サーバ群のログ情報管理支援の有効性に関する議論を行う．

以下，2 章では，従来のログ管理支援法について述べ，3 章では，本論文の先行研究であるサーバの依存関係抽出法について述べる．4 章では，本論文で提案するログ管理方式について説明し，5 章の実験結果に基づいて，6 章で提案方式の有効性に関する議論を行う．

2. 関 連 研 究

本章では，ログ情報管理支援に関するこれまでの研究および商用の統合運用管理ソフトウェアに関して述べる．

2.1 出現頻度を利用するログ情報の解析

文献 4)，5) では，複数の計算機や機器にログ収集管理サーバを配置して協調動作させ，ログの統一管理が可能な方式を提案し，ログ情報の解析に単語の出現頻度に基づく辞書を用いている．この研究では，連係動作機能，容易に機能の追加，変更が可能なログ収集解析サーバのプロトタイプシステムを実現している．

文献 6)–8) は，ログ情報にテキストマイニングを用い，異常事象はごく少数であるという理由から出現頻度の低い情報に着目した．これらの出現頻度に着目する手法は，大量のログ情報の中から異常を検出させる点ではきわめて有用である．しかし，個々のログ情報の意味に基づく方法ではないので，個々の障害の原因を詳しく調査する支援には利用しにくい．また，システム内の複数のログ情報の関連を調べることも難しい．

2.2 統合運用管理ソフトウェア

統合運用管理ソフトウェアとは，情報システムを効率良く運用するための機能を提供するソフトウェアであり，製品化も進んでいる[9)–11)]．

大規模で複雑なシステムにおいて，従来は分散管理していた様々な管理業務の一元化を可能にし，運用状況を 1 台のコンソールで把握できるので，管理者の負担を大幅に削減することができる．その統合運用管理機能の一部として，ログ情報の一元管理機能も備えている．

しかし，このログ管理機能は，各計算機および周辺機器ごとのログ情報を収集し一元化するにとどまり，システムを熟知し経験の長い上級管理者が，関連するログ情報を時系列ごとに手動で比較する手法や特定の語句やパターンで適合した箇所を調査する手法がとられている．これらの手法は，自分が担当している管理区域外について知識や経験の浅い初級の管理者にとっては，特定の語句やパターン自体を思いつくのが困難だという問題がある．また，上級管理者が設定して絞り込みが可能な場合であっても，現在着目している障害などの事象に関連した情報を順にたどる作業は従来どおりの手作業となるという問題がある．

3. サーバの依存関係抽出法

本章では，本論文の先行研究として我々が提案したサーバの依存関係抽出法[12)]について説明する．

近年，大規模なシステムを運用する組織においては，複雑なサーバ群を複数人の部署で集中管理する形態が増加している．このような組織のサーバ群は，ストレージサーバ，データベースサーバ，Web サーバ，各種アプリケーションサーバなどが，互いに依存関係を持つことが多い．その場合，一部のみを担当する管理者にとっては，全体システムの把握は困難であり，担当サーバ/サービスと依存関係があっても，さらにその先の依存関係先まで把握できない．

文献 12) で提案した方式は，各種サーバや周辺機器の設定情報を収集し，またシステム設計者が情報を補足する．それらの情報をもとに各部の依存関係をすべて抽出し，各管理者がシステム全体の構成を把握することの支援を行うものである．この方式を用いた構成管理支援システムの概要を**図 1** に示す．

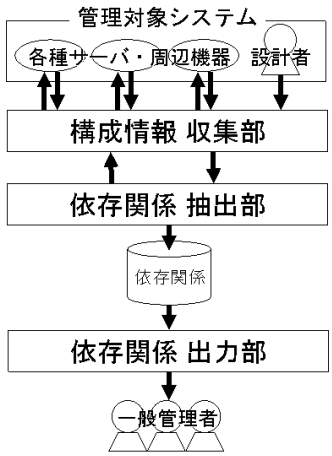

図 1 依存関係に基づく構成管理支援システムの概要[12)]
Fig. 1 The outline of configuration management system based on dependencies.

```
ASM2:search>>> depend -v %DIR%is14e1%/home/i2008 %DISK%*%*
       CLASS:DIR HOST:is14e1 PATH:/home/i2008
100:3  CLASS:DISK HOST:fs90 DISKname:/dev/dsk/c3t0d1s2
66:7   CLASS:DISK HOST:fs8-dctl0 DISKname:RG-01
76:8   CLASS:DISK HOST:fs8-dctl0 DISKname:/dev/dsk/c1t0d1s2
70:14  CLASS:DISK HOST:fs8-dctl0 DISKname:/dev/dsk/c1t0d7s2
ASM2:search>>> depend -vod 76:8
    CLASS:DIR HOST:is14e1 PATH:/home/i2008
<-- CLASS:DIR HOST:fs90 PATH:/home/fs5001 (nfs_mount)
<-- CLASS:DISK HOST:fs90 DISKname:/dev/dsk/c3t0d1s2 (ufs_mount)
<-- CLASS:DISK HOST:fs8-dctl0 DISKname:RG-01 (SAN FibreChannel)
<-- CLASS:DISK HOST:fs8-dctl0 DISKname:/dev/dsk/c1t0d1s2 (RAID5)
```

図 2 依存関係の出力例
Fig. 2 The example of output of dependencies.

たとえば，ディスクドライブなどの物理デバイス，RAID ディスクなどの仮想デバイスや HTTP プロトコルサーバなどの上位レイヤにおけるサービスなど，様々なオブジェクトに関して依存関係の調査を行う．調査には，それぞれのオブジェクトに関係する設定ファイル，管理コマンドの出力結果の解析や，場合によっては設計を行った管理者が与える情報をもとに行う．

本方式を用いた構成管理支援の試作システムの出力結果を**図 2** に示す．これは，我々の組織で実際に稼動している高可用性クラスタストレージサーバの構成情報の一部を入力として与え，NFS クライアント上のディレクトリを指定して依存している全ディスクの一覧と，その一覧の 1 つの詳細を出力させた例である．

ストレージに関する依存関係のみでなく，物理デバイスからアプリケーションまで広範囲にわたる間接的な依存関係の検索も可能である．たとえば，**図 3** に示すようなオブジェクトの依存関係が把握できていると，ストレージサーバの管理者は，自分が管理するとのディスクドライブ内のデータがどのような重要サービスに利用されているのか知ることができる．また，Web 上のアプリケーション管理者も，自分が管理するアプリケーションのファイルがどのサーバのどの物理資源を利用しているか容易に把握できる．すなわち，この方式によってレイヤを越えた依存関係の把握をも支援可能である．近年はストレージの仮想化も進んでおり，通常，このような依存関係は把握が困難であった．

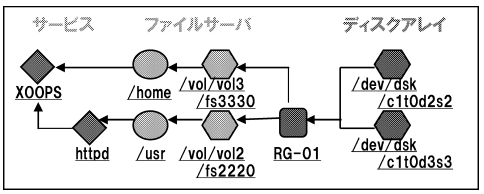

図 3 オブジェクトの依存関係例
Fig. 3 An example of dependencies among objects in servers.

4. サーバの依存関係に基づくログ管理支援法

本章では，前章で述べたサーバ群を構成するオブジェクトの依存関係情報を基にログ情報の管理を支援する方式について説明する．

4.1 概　　要

サーバには様々なログファイルが保存されており，また，1 つのファイルにも様々な部位に関する情報が記録される．すなわち，ログファイルの各行は何らかのオブジェクトに関係する情報を表示している．そこで，**図 4** に示すように，各ログ情報とオブジェクトの対応を求める．以下，本論文ではこれを**リンク**と呼ぶ．図 4 では縦方向に伸びる太い双方向矢印がリンクである．リンクは，ログファイルの単位ではなく，4.3 節で述べるように 1 事象に関する情報ごとに作成する．一方，図 4 で横方向に伸びる細い片方向矢印は 3 章で述べたオブジェクト間の依存関係である．このリンクおよびオブジェクト間の依存関係情報を利用すると，別ファイルや別サーバの情報であってもシステムが関係をたどることができ，管理者が現在注目しているログ情報に関連のあるログ情報を，自動的に抽出して提示を行う．たとえば，Web サーバのエラー情報を発見した管理者が，そのログ情報を入力として与えて問い合わせると，その症状が発生したファイルが格納されているファイルサーバで発生した該当ディスクの故障に関するログ情報を表示することができる．

この方法に基づく管理者支援システムの構成を**図 5** に示す．図中右上の部分は 3 章の図 1 で述べた部分

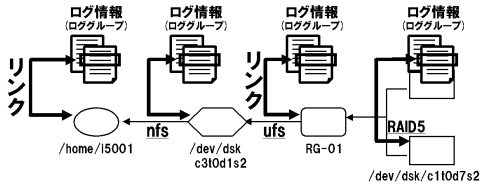

図 4 オブジェクトとログ情報のリンク
Fig. 4 Links between object and log information.

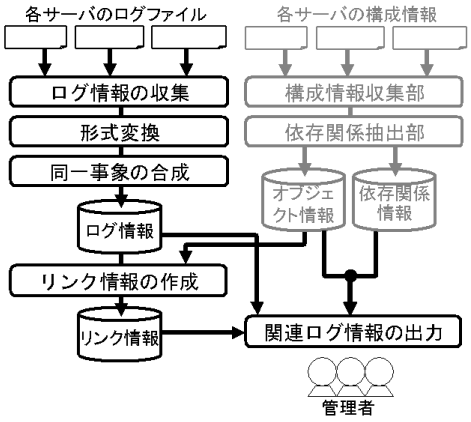

図 5 ログ情報のリンクを用いた管理者支援システムの構成
Fig. 5 Overview structure of management system based on log links.

である．それによって得られたオブジェクト情報と収集されたログ情報をもとにリンク情報を作成し，管理者からの問合せに応じて，システムがリンク情報と依存関係情報をたどることで関連ログを発見し，出力することができる．各部の詳細に関しては次節以降で述べる．

近年の大規模な情報システムを管理する現場では，複数人の管理者が分担して管理する組織が増えている．各管理者は自分が担当する部位のログファイルは頻繁に参照し，様々なログ情報の解釈に慣れている．しかし，障害の解析時などに他の管理者の担当部位が関係することがあるが，通常参照していないログファイルの解釈は困難であり，また膨大なログ情報の中に参照すべき情報が存在していることに気がつくのも難しい．本章で述べる方式を用いることで，そのような管理者の作業を支援することができる．

4.2 ログ情報の収集

ログ情報収集機構は，次節で述べるリンク情報生成機構からログ情報の収集に関する要求を受け取り，対象サーバからログ情報を収集し，結果を返す．

また，ログ情報は syslog や apache など，OS やアプリケーションの種類やバージョンによって形式が異なる．そのため，統一形式に変換することでその差異を吸収する．実際の形式は実装の環境によって異なるが，5 章で述べる試作システムでは Solaris の syslog 形式に統一し，日付，ホスト名，ログ ID やメッセージ本文などを判別して処理可能にしている．

4.3 リンク情報の生成

ここでは，各ログ情報をその情報が関係しているオブジェクトとリンクさせる方法について述べる．

システム内で発生したすべてのログ情報について，システムを構成する全オブジェクトとの適合を検査することで，リンクの有無を決定する．各オブジェクトには，それを一意に識別することが可能な値の組（キー）が定義されており[12]，それらの文字列がログに含まれているかどうかで判断を行う．一般に，ログ情報の 1 行に含まれる情報はわずかであるが，1 つの事象に関係して複数行が出力されることも多い．また，複数の事象が混ざり，同一事象の複数行が連続しない場合もある．同一事象の判断は，ログの各行に含まれるログ ID，プロセス ID，プロセス名や時刻などの情報をもとに行う．前述のオブジェクトとの適合検査など，本論文で扱うログ情報の最小単位は同一事象に関する複数行をまとめたものである．複数行をまとめる方法は，OS やアプリケーションなどのログの形式に依存しており，我々の試作システムについては 5 章で述べる．

オブジェクトを識別するキーが複数の値であるとき，1 事象のログ情報にその一部しか出現しないこともある．そのようなログ情報を**不完全なログ情報**と呼ぶ．たとえば，4 つの値のうち 3 つしかなかった場合には，**関連度 75%**と判断し，リンクの確からしさとして記録する．また，このように不完全なログ情報の場合には複数のオブジェクトに適合する可能性があるが，それらの候補オブジェクトすべてへのリンクを記録する．たとえば，NFS クライアントのメッセージに “`NFS server host1 not responding`” と出力された場合，サーバのホスト名は分かるがどのファイルシステムが原因で発生したのか不明である．この場合には，このクライアントが host1 へ NFS マウントしている全ファイルシステムがリンク対象オブジェクトとなる．

ここで，システム全体のオブジェクト数を n とし，全ログ情報の事象数を m とすると，適合性検査の計算量は mn に比例する．ただし，オブジェクトのクラスによって，キーとなる属性の数が 2〜4 個と異なるため，これによっても変動する．大規模なシステムでは，数千以上のオブジェクトが存在し，膨大な事象数のログ情報が出力される．これらのすべての組合せについて適合性を検査するには大きな計算時間となる．

そのため，計算時間削減のため以下のように処理する．

- ログ情報はファイルの末尾に追記していく方法で単調増加し，出現した情報が消えることはない．定期的に前回との差分情報のみを検査する．管理者が利用する際には，前回の定期検査時からその時点までの情報のみを処理する．そのため，ログファイルごとに検査が完了した最終行番号を記録しておく．また，管理者の問合せ処理時には，管理者の問合せ箇所から依存関係があるオブジェクトのみを検査することで，応答時間の短縮が可能である．

ただし，複数行にわたる同一事象の途中で検査が行われることもありうる．そのためには最終行番号よりある程度遡った範囲まで同一事象を探し，境界にまたがる合成が発生した場合にはそのログ情報に関するリンクの張りなおし処理が必要である．

なお，この手法は総計算時間を削減するものではない．定期的に差分を計算しておくことで，問合せ時の計算時間を削減することが目的であるが，問合せがまったくなくても計算が行われるという欠点もある．

- オブジェクトとログ情報のすべての組合せの中には，管理者の知識から明らかに関連がないことが分かっている場合が少なくない．たとえば，HTTP サービスのサーバプログラムなど，専用ログファイルに情報を出力する場合は，他のすべてのログファイル内のログ情報とこのサービスのオブジェクトの間の検査は不要である．

そのため，管理者が持っているこのような知識をもとに，オブジェクトとログ情報を分類し，その対応をあらかじめ決定しておく．対応表の例を**表 1** に示す．複数のアプリケーションやデバイスなどに関するログ情報が 1 つのファイル内に混在するが，単純なキーワードの有無により大別することができる．そのように分類したログ情報の各集合を**ロググループ**と呼ぶ．一方，オブジェクトは Disk や Service などの**クラス**に分類されており[12]，そのクラス名や一部の属性値を指定することでオブジェクトを分類することができる．

表 1 において，ロググループの指定は “**ログファイル名**” または “**ログファイル名**.**パターン**” となっている．各ログ情報について対応表の各行を参照し，ファイル名が一致するか，パターンがあればその文字列がログ情報に含まれているか調べ，一致した場合は対応するオブジェクトを検査対象とする．大規模なシステムではオブジェクトの数が 1 ホストでも数千個以上となることが想定されるので，分類に関する設定が 1 つのログファイルに数十パターンあった場合でも，適合性検査の比較回数を大幅に削減可能である．

表 1 ロググループとオブジェクトの対応表の例
Table 1 An example of table of correspondence between object groups and log groups.

オブジェクトの種類	ロググループ
Service.sendmail	syslog.sendmail
Service.httpd	messages.httpd access_log error_log

一例として，ログ情報が 10 種類に分類され，各々が全ログ情報の 10%ずつと仮定する．また，各ログ分類に対応するオブジェクトも 10 種類に完全に分けることができ，各々が全オブジェクト数の 10%ずつと仮定する．この場合は，10 種類の各分類の中でログ情報とオブジェクトの全組合せを検査しても，全体のログ情報とオブジェクトの全組合せを検査する場合と比較すると，計算時間は 10%に削減できる．

5. 試作システムによる実験

前章で述べた方式に基づく試作システムの実装と，これを用いた実験について説明する．

5.1 試作システムの実装

著者らの組織では，大規模できわめて複雑なサーバ群の運用を行っているが，これらのサーバ群のログファイルをそのまま実験に用いると，たとえば，実際のユーザの所有するファイル名がログに記録されていることもあるので，そのような情報を実験担当者が知ることとなる．すなわち，セキュリティやユーザの個人情報保護の理由により，運用中のシステムを直接参照する実験は実施できない．しかし，著者の一部はこれらのサーバ群の設計や運用に直接携わっているため，実際のシステム設計書を参照して構成情報の一部をそのまま入力として与え，障害事例などの実際の運用経験に基づいた実験を行った．

試作ツールの実装およびすべての実験は Solaris10 が稼動する 3 台の Sun ワークステーション上で行い，リンク情報生成機構の記述には Perl を用いた．ログ情報収集機構は，ログ管理サーバが管理している全サーバにアクセス・認証して，安全にログファイルを収集する機構であるが，今回の実験では各ワークステーションの全ログファイルを手作業でコピーしたため，今回は収集部を実装していない．収集後の統一形式への変換には Perl スクリプトを作成し，Solaris10 の syslog 形式に統一した．

各ログファイルはリンク情報生成の前処理として同一事象に関する複数行をまとめる処理を行う．今回の試作システムでは Solaris10 の syslog 形式に合わせて，

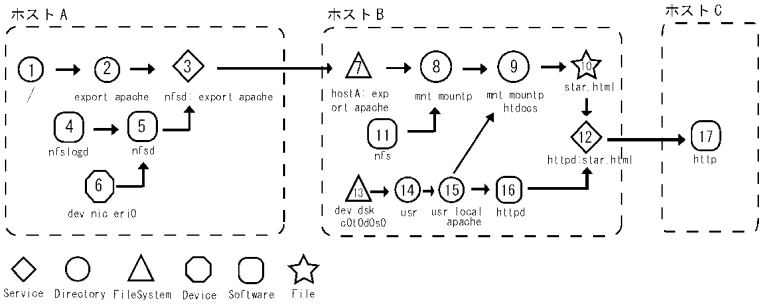

図 7 Web サービスを行う試験システムのオブジェクト依存関係
Fig. 7 Dependencies of objects in web service system.

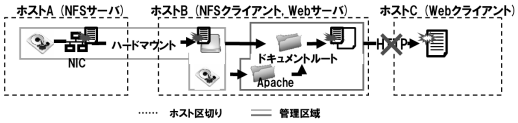

図 6 Web サービスを行う試験システムの概略
Fig. 6 Configuration of experimental system of web service.

ログ情報のタグ，ログ ID，プロセス名，プロセス ID がすべて一致するものを同一視した．また，sendmail のログに関してはメッセージ ID を用いて分類し，いずれの場合も同一事象が連続行でなくてもまとめる．さらに，syslog では同一内容が繰り返された際に “`last message repeated`” と出力されるが，この場合は内容が一致するわけではないが例外的に行の合成対象とした．

実験内容は，実際のサーバで起こりうる状況を想定したものであるが，本稼動しているサーバは千本以上のディスクドライブからなるディスクアレイを含むなど，ハードウェア構成は異なる．一部のオブジェクトは架空のものを作り，ディスクアレイのログも本サーバのログを模倣して手作業で作成した．一方，ハードウェアと直接関連しないサービス類は，NFS，HTTP，SSH などを実際にサービスさせ，実際のログファイルに出力された情報をそのまま用いた．

実験が想定しているシステムはオブジェクト数が数千以上になる大規模なサーバであるのに対して，実際に実験を行ったホストはオブジェクト数が数十であるが，本論文の方式は，オブジェクト間の依存関係ではなく，オブジェクトとログ情報のリンクに関するものであるため，縮小された環境でも動作確認が行え，また大規模サーバで利用する状況を推測できた．

5.2 障害発生実験

図 6 に示すような構成のシステムを用いて実験を行った．これは，サービスプロバイダや大企業などにおける Web サービスに関するシステムを想定し，簡略化したシステムである．ホスト A はディスクアレイ装置を持った NFS ファイルサーバであり，そのファイルシステムをホスト B が NFS マウントしている．また，ホスト B で稼動している apache を用いて，ホスト C 上の Web ブラウザからファイルにアクセスしている．このシステムを用いて，運用中にディスク装置を停止させたり，ネットワークを切断したり，意図的に障害を発生させたりした場合の挙動を調査した．

この試験システムにおいて，障害に関係するオブジェクトの構成を**図 7** に示す．

ホスト A の Ethernet ケーブルを切断すると，ホスト A の messages の 356 番目に “`eri0: No response ...`” というメッセージが出力される．NFS がソフトマウントであった場合は，ホスト B の messages の 224 番目で “`NFS getattr failed for server hostA ...`” と出力される．一方，Web サーバの apache でもファイルがオープンできなかったために error_log へ情報が出力される．これらのメッセージに関してリンクを作成した結果を**表 2** に示す．なお，この実験では実際の`/var/adm/messages` ファイルなどを収集しており，上記以外のログ情報も存在するが，表 2 に示したリンクは，図 7 のオブジェクトのみを用いたこの実験に関するリンクのみである．

また，NFS がハードマウントであった場合には，ホスト B の messages で “`NFS server hostA not responding still trying`” と出力され，apache は open システムコールでブロックされるために，error_log へは何も出力されず，ネットワークが復旧した時点で open できるのでその後に access_log が記録される．

6. 議　　論

本章では，前章で述べた実験結果をもとに，サーバ各部の依存関係に基づくログ情報管理法に関する議論を行う．

表 2 ネットワーク障害実験で作成されたリンク情報
Table 2 Log links for network trouble on web service system.

リンク ID	オブジェクト ID	ホスト名	ログファイル名	ログ事象番号	出現した属性値	関連度 (%)
1	6	hostA	/var/adm/mesasges	356	hostA，eri0	100
2	4	hostA	/var/nfs/nfslog	42	hostA，nfslogd	100
3	11	hostB	/var/adm/messages	224	hostB，nfs	100
4	16	hostB	/usr/local/apache/logs/access_log	14	hostB，star.html	100
5	12	hostB	/usr/local/apache/logs/error_log	118	hostB，star.html	100

6.1 不完全なログ情報の場合

表 2 に示した例では，必要なすべての属性値がログ情報に含まれており 100%の関連度でリンクが生成できたが，すべての場合にリンク先オブジェクトが明確に決まるとは限らない．ログ情報に属性値の一部のみしか書かれていない場合について述べる．

5.2 節の実験で，ホスト B の messages に出力された NFS のエラー “`NFS getattr failed for server hostA ...`” には相手のサーバ名も書かれるが，エラーの原因となったファイルシステムは不明である．対話型のコマンドラインで発生した場合には明らかであるが，ログ情報のみを参照しても分からないため，どのサービスに影響が出たのか管理者が判断するのは難しい．近年はログ情報が詳しくなる傾向があり，このような場合にファイルシステム名が書かれる OS もあるが，一般的にログファイルには不完全な情報しか出力されないことも多い．この実験でも，図 7 のオブジェクト 11 にはリンク可能であるが，ファイルシステムやディレクトリのオブジェクト（ID 7，8，9）などに直接リンクすることができなかった．

図 6 では 1 つの NFS マウントについてのみ図示しているが，複数のファイルシステムをマウントする可能性がある．その場合，ID7，8，9 と同様な一連のオブジェクト群も各々に存在するが，NFS デーモンは共通であるため，ID11 のオブジェクト 1 つから ID8 などの各マウントポイントのディレクトリに依存関係が構築されていることになる．すなわち，NFS 関係のすべてのログ情報が NFS デーモンにリンクされるため，依存関係をたどって表示した際には，他のマウントポイントに関するログ情報と混ざって出力される．

しかし，上記ログ情報の文字列からホスト A の NFS サービスに関係していることは分かるので，リンク情報の生成時にオブジェクト 7 にも関連度の低いリンクができる．ホスト A の他のファイルシステムもマウントしている場合には，オブジェクト 7 と同様のそれら複数のオブジェクトともリンクされる．この情報のみでは，複数の候補から選択するのは管理者の作業となるが，たとえば，障害の原因がホスト A 側の 1 つのディスクドライブであった場合には，そのエラーメッセージがデバイスのオブジェクト（図 7 では省略）とリンクしているため，システムがオブジェクト間の依存関係をたどった際に，関係オブジェクト中でエラーとリンクしているオブジェクトが 1 つであれば，ホスト B 上で候補となっている複数のファイルシステムのどれであるか推測することも容易である．

すなわち，この例のように不完全なログ情報であっても，ホスト B の NFS エラーとホスト A のディスクドライブのエラーの間に関連がある可能性をシステムが管理者に自動提示できる．

以上のように，ログ情報が不完全な場合など，管理者が複数のサーバにまたがって障害の原因究明を行うことは少なくない．通常，これは管理者の経験が影響するような困難な作業であるが，本論文の方式で支援することによってそのような作業を省力化可能である．

6.2 サーバシステムの規模による有用性

一般に運用されているシステムの規模を構成の複雑さとシステムが扱うデータ量の 2 つの観点から考えると，以下の 4 種類のシステムに分類できる．

なお，本論文の方式はシステムの稼動を直接支援するのではなく，管理者という人間の作業を支援する方式であるため，システムの構成上の技術的特徴に基づいて分類を行っているのではなく，システムを扱う管理者の視点で論じる．そのため，ここでの分類は，任意のシステムが 4 種類に明確に分けられるという意味ではなく，各システムを扱う管理者の経験などによっても分け方が異なる可能性がある．

小規模システム 小さい組織や，一部の部署の構成員のみが利用するシステムでは，サーバの台数も少なく単純な構成で複数のサービスに兼用しており，扱うデータ量も少ない．このようなシステムではログ情報も少なく，1 人または少人数の各管理者がシステム全体の構成を把握できるため，ログ管理作業の支援機構が不十分でも必要な情報の埋没化が発生しにくい．

データ量の多い大規模システム 多数のユーザの情報

を扱うシステムでは扱うデータ量が多くなる．たとえば，人数の多い組織のメールサーバやファイル共有サーバでは，メール/ファイルの数や総容量が大きくなる．そのため，それらを保持するために，ディスクドライブ数やインタフェースカード数が増え，システムの部品数が大きくなる．

しかし，ある Web サービスやデータベースサービスなどに特化したシステムの場合には，扱うデータが膨大でハードウェア部品数が増えても，システムの基本的構成は単純である．このようなシステムでは，データの処理量や部品数に比例してログの量が増加するが，同種の情報の発生頻度が高いためであり，検索機能や自動通知機能など単純な支援ツールを用いることでログ管理作業の効率化が行える．

複雑な構成の大規模システム　近年は，多数のサービスが情報システム上で提供されているが，バックエンドにデータベースを必要とするサービスも多く，その際に個々のサーバごとに独立したデータベースを持つのではなく，資源の有効利用，信頼性向上，管理作業の省力化を考慮して，信頼性の高い 1 つのデータベース専用サーバに集中させる構成がとられる．また，大容量のストレージシステムが安価に構築できるようになり，様々なサーバが扱うデータを一元管理する構成のシステムが増えている．このような場合には，個々のサービスが独立したシステムではないため，各サーバが複雑な依存関係を持った構成となる．データ量が極端には多くない場合でもある程度の部品数があるシステムで多様なサービスを行う場合には，このように複雑な構成のシステムが今後ますます増加すると考えられる．このようなシステムを複数人で管理する際のログ管理には本論文で提案する方式がきわめて有効である．

また，近年のストレージシステムでは，様々なポリシのデータを集めた際のさらなる資源の有効利用やデータ移行の容易性のために，仮想化を行う技術が注目されている．仮想化の利点は大きいが，管理者の作業の面から考えると，上位層の各サービスと下位層のハードウェア部品の依存関係の把握ができず，ログ情報の参照がきわめて困難になる．先行研究[12] のシステムでは仮想化機構の設定ファイルを入力することで依存関係が把握できるため，本論文の方式を用いた関連ログ情報の抽出も容易である．

複雑な構成でデータ量の多い大規模システム　扱うデータ量が多いシステムにおいて，高信頼性サーバを採用したり，複数サーバにまたがる依存関係を持ったりするシステムも増えている．この場合でも前項同様に本論文の方式が有用である．

様々なソフトウェアのログ情報が混ざり，また個々のソフトウェアが出力するログ情報の行数が膨大になるため，管理者が手作業でログを見ることは非常に困難であり，提案方式の有用性がきわめて大きくなる．

以上のように，複雑な構成の大規模サーバでは本論文の方式の有用性が高いが，実際に稼動しているサーバでの大量のオブジェクトと膨大なログ情報の処理に要する時間に関して，5 章の小規模な実験環境では，オブジェクト数もログ量も少ないため評価が得られていない．様々な環境における実サーバの構成情報・ログ情報を用いた評価は今後の課題である．

6.3 管理者の階層と管理者教育

著者はこれまで大規模システムの運用管理に携わっており，その間に大勢の若い管理者を教育してきた．また，信頼性向上や性能向上のため，新しい構成法のシステムを多数設計してきた[2]．そのような複雑な構成のシステムで障害が発生すると，各システムベンダのエンジニアも未経験な新たな障害であるため，その障害解析作業も指導してきた．その経験から，システム管理経験の短い管理者の特徴の 1 つとしてログ情報の参照・活用が十分でないことが分かった．

ログファイルには，システムを構成する様々なハードウェア部品や様々な基盤ソフトウェアがエラーメッセージやステータスを出力しており，障害解析時には有力な手がかりとなる．しかし，複雑なシステムでは膨大な行数の情報が出力される．経験年数の長い管理者（上級管理者）であれば，経験に基づいて比較的短時間で必要な情報を収集できるが，経験年数の短い管理者はログの調査を十分に行わず，直接的症状や便利な管理ツールの実行結果にとらわれる傾向がある．特に，根本原因箇所が障害発生箇所と異なる場合には原因究明に多くの時間を費やす．その結果，上級管理者がシステム管理の現場から離れられないという問題にもつながっている．

近年の情報システムは大規模化・複雑化しており，今後もますます複雑化していくと，それに従って障害も複雑化していくため，上記のような問題には多くの組織で直面することとなる．そのような場で本論文の方式を活用することにより，中級以下の管理者のログ解析を可能とし，また上級管理者であっても困難な作業の効率化が行える．

一方，初級中級の管理者の教育について考えるとき，大規模で複雑なシステムは障害の複雑さが適切でなく，管理者教育が難しくなっていることも，著者の経験で問題となっていた．本論文の方式によって障害に関連

するログ情報の絞込みが可能であるが，ログ情報の不完全さから完全な絞り込みができるとは限らない．しかし，それらの候補の存在箇所が提示され，そこからさらに絞り込みを行うという作業は，初級中級管理者にとって最適な教育材料となりうる．

7. おわりに

本論文では，サーバの各部品や様々なサービスの間の依存関係に注目し，システム全体の全ログ情報の中から管理者が現在注目している事象に関連のあるログ情報を担当区分を越えて自動抽出する方式を提案した．

近年はストレージシステムも大規模化しており，大企業や大学の情報システムは大規模サーバ群による集中管理を行う方式に移行しつつある．そのような大規模で複雑なシステムの管理には，全体を把握する 1 人のエキスパートではなく，複数人の管理者が分担して作業する組織が増えている．

ログ情報の参照は管理者が行う重要な作業の 1 つであるにもかかわらず，1 人で全体のシステム構成を把握しておらず，複数人で分担管理を行う現場では，これまでの手法による支援が困難であった．本論文で提案する方式の利用によって，大規模な情報システムの管理作業が効率的に行える．すなわち，現在よりもさらに大規模なシステムの構築も可能となる．

今回行った実験はほとんどを Solaris 上で行ったが，システムによってメッセージの形式やオブジェクトの構成が異なる．今後は，異なるオペレーティングシステムや様々な形態のストレージサーバなどのログ情報を用い，汎用性のある環境で実験を行い，計算時間の定量的評価も行うことで有用性の確認を行う．

参考文献

1) 敷田幹文，井口 寧，丹 康雄，松澤照男：大規模分散システムの集中運用管理における効率化技術の提案，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，pp.75–80 (1999).
2) 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，pp.57–62 (2001).
3) 敷田幹文，井口 寧，藤枝和宏，松澤照男：高可用性システム統合監視機構の提案，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，pp.51–56 (2002).
4) 神尾正和，石田常竹，箱田貴久：分散データベースを用いた大規模ログ管理システム，電子情報通信学会データ工学ワークショップ論文集，pp.1–8 (2005).
5) 神尾正和，石田常竹：ログの統一管理及び異常検出に関する研究，情報処理学会コンピュータセキュリティ研究報告，pp.77–82 (2004).
6) 江端真行，小池英樹：不正侵入調査を目的とした複数ログの時系列視覚化システム，情報処理学会論文誌，Vol.47, No.4, pp.1099–1107 (2006).
7) 高田哲司，小池英樹：見えログ：情報視覚化とテキストマイニングを用いたログ情報ブラウザ，情報処理学会論文誌，Vol.41, No.12, pp.3265–3275 (2000).
8) 高田哲司，小池英樹：ログ情報視覚化システムを用いた集団監視による不正侵入対策手法の提案，情報処理学会論文誌，Vol.41, No.8, pp.2216–2227 (2000).
9)（株）日立製作所：JP1 Version 8 (2007). http://www.hitachi.co.jp/Prod/comp/soft1/jp1/
10)（株）日立製作所：JP1 Version6 JP1/Base マニュアル (2001).
11) 富士通（株）：SystemWalker/CentricMGR Version 13.2 (2007). http://systemwalker.fujitsu.com/jp/
12) 森 一，敷田幹文：サーバの依存関係を考慮したシステム構成管理の支援法，情報処理学会論文誌，Vol.46, No.4, pp.940–948 (2005).

(平成 19 年 6 月 11 日受付)
(平成 19 年 12 月 4 日採録)

敷田 幹文（正会員）

1965 年生．1995 年東京工業大学大学院理工学研究科情報工学専攻博士後期課程修了．博士（工学）．同年北陸先端科学技術大学院大学情報科学センター助手．2001 年同助教授．大規模分散システム，グループウェアに関する研究に従事．ACM，電子情報通信学会，日本ソフトウェア科学会各会員．

後藤 宏志

1982 年生．2007 年北陸先端科学技術大学院大学情報科学研究科博士前期課程修了．同年ヤフー（株）システム統括部サイトオペレーションズ部．大規模ネットワークの構築・運用に従事．