ログ情報の解析に基づくネットワークサービス監視システム*

3R－10

犬束 敏信　浜田 雅樹**
***

1．はじめに

本稿に於いて筆者らはログ情報の「見張り」と「解析」に基づくネットワークサービスの監視モデルを提案する．

我々は，ネットワークサービスを監視するための情報源として，サービス自身が生成するログ情報に着目した．サービスにより生成されるログには，サービスの実行状態を示すさまざまな情報が含まれている．これを適切な手順で「解析する」ことにより，利用状況の把握，障害や不正利用の発見が可能となる．ログの解析手順の記述手法として，従来のようなプログラミング言語を用いず，データフロー図による図的記述を導入した．これにより監視プログラムを生成する負担が軽減できた．また，本システムではリモートホストからログを定常的かつ自動的に「見張る」機構を持ち，多ホストに散在するログの集約監視，および障害や不正利用の早期発見機能を実現している．さらに統計的な解析も可能であり，容易に利用状況が把握できる．

2．ネットワークサービス監視へのニーズ

インターネット，イントラネットの利用者数は日々増加しており，WWW，電子メール，NetNewsといったサービスは既に不可欠な存在である．本稿では，ネットワーク接続されたホスト上で動作するプログラムの協調動作により実現されているサービスを「ネットワークサービス」と定義する．ネットワークサービスを正常に運用するために，個々のサービスが正常に動作していることを定常的に確認し，かつ障害や不正利用があった場合にはすぐにこれを検知する必要がある．これがサービスの監視である．

管理の標準に関しては，ネットワーク機器の場合，既に管理プロトコルの標準が確立しており[1]，殆どのネットワーク機器に実装されている．これに対し，ネットワークサービスの管理には標準と呼べるものがまだない．

さらに管理用ツールに関しても，ネットワーク機器にはOpenView[2]などの製品が標準的に用いられている．ネットワークサービスの管理にも，いくつかのツールが製品化されてはいるが[3,4]，これらは特定のサービスだけしか管理できない，あるいは汎用のサービスを対象としても通信ポートやプロセスのヘルチェックしかできないなど，まだまだ実用には不十分と言える．

この現状で，管理者は独自でスクリプトを記述するなどの方法により監視を行っている．このため，サービスの監視には経験豊富な管理者の知識が要求され，非熟練者による監視は困難であった．また，サービスの健常性を定期的にチェックすることは管理者の大きな負担であり，さらに，サービスの種類の増加に伴い管理者の負担はますます大きくなってきた．このような背景から，ネットワークサービスの監視を支援するシステムの実現が望まれていた．

3．提案するシステムの特長

本モデルは，サービスが生成するログを「見張る」「解析する」ことに基づいている．これらを実現するために，

a) Javaエージェントによる分散監視機構

b) データフロー図（以下，DFD）による図的記述手法

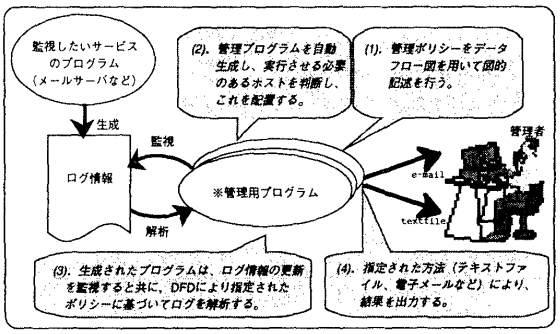

図1．システムの動作原理

を用いている．これにより管理者は，a) リモートホストからログを定常的かつ自動的にチェックすることにより，障害や不正利用の早期発見，各ホストに散在するサービスの監視の連携が可能となる，b) 解析手順は部品の組み合わせで生成されるDFDを記述するだけ指定できるため，特別なプログラミングの知識が必要ない，という恩恵が得られる．また，統計解析機能も具備しており，サービスの利用状況の把握も容易にできるようになった．

実際に利用者がすべき操作は，(1)DFDにより解析手段を記述することだけである．以降は，システムが(2)DFDをエージェントへと変換し，必要な場所に配置する．(3)これらのエージェントが，ログ情報の更新の監視し，指定された方法でログ情報を解析，(4)解析結果を管理者に通知する，のステップを順次実行し監視を行う（図1）．

4．システム概要

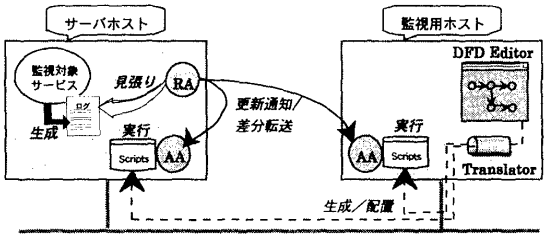

図2．システムの構成図

本システムの構成を図2に示す．この図では監視用ホストと監視対象ホストを別に記述しているが，もちろん同一のホスト上での動作も可能である．本システムは，

(1) DFDエディタ（DFD editor）

(2) トランスレータ（Translator）

(3) 実際に監視を行うエージェント（RA，AA）

により構成されている．

4.1 DFDエディタ

管理者は，DFDエディタを用いて監視内容をDFDにより記述する．DFDは，ノード（操作）とアーク（データの流れ）から構成され，データへの操作手順の流れを直観的に記述できるという特長を持つ．ライブラリとして用意されたノードの組み合わせにより記述するため，特別なプログラミング言語の知識も必要なく，初心者でも訓練は必要ない．監視の対象となるログもノードで抽象化することにより，ホスト間を跨ぐような解析も容易に記述できる．

* "A study on network service monitoring system based on analysing log".
** Toshinobu INUZUKA, Masaki HAMADA
*** NTT Software Laboratories

4.2 トランスレータ

記述されたDFDはトランスレータにより，実行可能なプログラムを含むエージェントへと変換される．DFD上の各ノードは，それぞれ１つのエージェントへと変換される．生成されたエージェントは適切なホストへと配置され，これらのエージェントの協調によりネットワークサービスの監視を行う．

4.3 エージェント

トランスレータにより生成されるエージェントには２種類ある．一方は，ログを「見張る」ためのものである（図1中のRA）．これは，ログの更新を検知し後述するAAへと通知する役割を果たす．RAにより障害や不正利用を迅速に発見することが可能となる．他方は，「解析」を行う，すなわち，ログを処理したり，処理結果を管理者へ通知するためのものである（図1中のAA）．

5．評価

ネットワークサービスは様々なOS上で提供されているため，監視システムも可搬性を持つことが望ましい．また，今後ますますサービス種の増加が予想され，これに対応するためには，高い拡張性を持つことがも必要条件である．これらを踏まえてプロトタイプ実装を行った．

5.1 実装例

本システムをJava（一部Perl）を用いて実装した．Java，Perlといった可搬性の高い言語を用いることにより広いプラットフォームでの動作が可能となった．

DFDを構成するノードは，JavaBeansを用いて実装を行った．JavaBeansを用いることにより高い拡張性が実現できた．すなわち，(1)各管理サイトに合わせて，必要に応じて容易にノードを増減できる．(2)新サービスの追加等により既存ノードだけで対応ができなくなった時にも，既存システムに何らの変更もなく新たなノードの追加が可能である，という特徴を持つ．これにより要求の変化に対して，柔軟かつ迅速な対応が可能となった．現在までに，18種のノードを作成している．

監視エージェント（RA）は，正規表現を用いたファイル名により指定されたログファイルを監視する．これにより，運用時にファイルがローテートされ複数のファイルに分かれている場合にも，仮想的にひとつのファイルであるかのように扱うことができる．また，毎回の実行時には，ファイル全体だけでなく，前回実行時から増加したログだけを対象にすることも可能である．さらに，ログの代わりにプロセスの出力結果を用いることも可能である．

5.2 適用例

筆者らは，幾つかのネットワークサービス監視に対して本システムを適応しその有用性を確認している．ここでは，協調管理環境において，NMSが出すトラップをオペレータがどのように処理しているかの状況を追跡するサービスの監視に適応した例を示す．この監視対象サービスでは複数のNMSによる協調管理を考慮している[5]．

このシステムの監視では，

・各NMSがそれぞれいくつのトラップを受け取ったか
・各NMSがそれぞれいくつのトラップを処理したか
・トラップが発生してから，処理されるまでに要した時間

などの項目が興味の対象となる．これらの項目を解析するために，図３の例のようなDFDを記述した．これらのDFDを変換することによって得られたエージェントにより，定期的な監視がなされ，図４の例のような出力が定期的に報告されている．また，5.1節で述べたプロセスの出力結果の解析から，サービスを構成するプログラムの動作確認も行うことが可能である．

6．さいごに

6.1 まとめ

本稿で，筆者らはネットワークサービスを監視を支援するシステムを提案した．本システムはサービスが生成するログ情報を基にしており，個々のサービスに対応した柔軟な解析を行うことが可能となる．さらにリモートホストからログを定常的かつ自動的にチェックする機構を実現したことにより，障害や不正利用の早期発見，各ホストに散在するサービスの監視の連携が可能となった．これらにより管理者の負担が大きく軽減できた．

6.2 今後の課題

ログ情報の解析結果を管理者へと通知するだけでなく，結果をサービス自身へフィードバックする機構を取り入れる予定である．これは監視だけでなく管理を行うシステムへの拡張を意味するものである．

また，システムを構成するエージェントをモーバイルエージェントとして作成し，柔軟なデータ収集機構，CPU負荷やトラヒックに応じた最適配置機構を取り入れていく予定である．

参考文献

[1] Marshall T. Rose, "The Simple Book: An Introduction to Internet Management, 2nd Edition," PTR Prentice Hall, Inc., 1994
[2] "HP OpenView", http://www.openview.hp.com/
[3] "SiteLog/Analyzer", http://www.isac.co.jp/Sitelog/
[4] "WebTrends Log Analyzer", http://www.webtrends.com/products/cartridge/log.htm
[5] 浜田, 犬束, et al. "インターネットにおける協調管理プラットフォームの提案", 情報処理学会第10回会分散システム運用技術研究会, 1998.

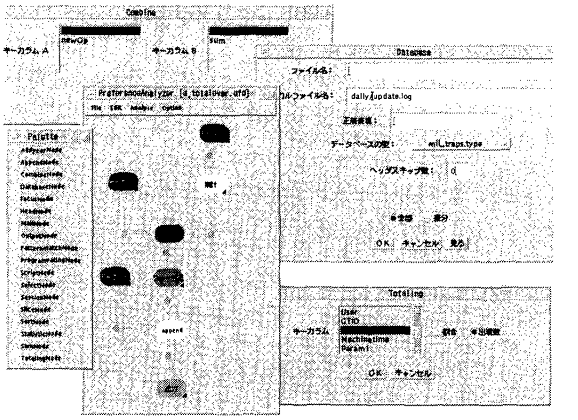

図３．記述例

```
--==################################=---
      MII Weekly Report
--==################################=---
Startingdate:
1998 Nov 29
Endingdate:
1998 Dec 05
---------------------[NEW TRAPS - TOTAL]--
N 312
-------------------------[TRAP SOURCES]--
10.2.1.2 28
10.2.3.1 28
10.2.3.2 256
------------------------[TRAPS PER DAY]--
Day      number of traps
-----------------------
>>NMS angels
Day 30      Traps  11
Day 01      Traps  23
Day 02      Traps  11
Day 03      Traps  13
Day 04      Traps  30
Day 05      Traps  44
Day 06      Traps  24
>>NMS gruner
Day 30      Traps  21
Day 01      Traps  13
Day 02      Traps  26
Day 03      Traps  18
Day 04      Traps  20
Day 05      Traps  34
Day 06      Traps  24

----------------------[OVERVIEW - TOTAL]--
new           312
accept(self)  35
accept(sync)  29
delete(self)  55
delete(sync)  82
--------------------[OVERVIEW - PER NMS]--
>>NMS angels
new           156
accept(self)  25
accept(sync)  10
delete(self)  41
delete(sync)  23
>>NMS gruner
new           156
accept(self)  10
accept(sync)  19
delete(self)  14
delete(sync)  59
--------------------[STATISTICS - TOTAL]--
Trap closing time (average)    :  5927 (m)
Trap closing time (maximum)    :  13639 (m)
Trap closing time (minimum)    :  8 (m)
------------------[STATISTICS - PER NMS]--
>>NMS angels
min 33 (m)
avg 5836 (m)
max 13639 (m)
>>NMS gruner
min 8 (m)
avg 19 (m)
max 333 (m)
```

図４．出力例