コログ：ログの共有化によるディペンダブルシステム構築に向けて

石山　直毅† 　中田　晋平† 　小林　聡† 　倉光　君郎††

† 横浜国立大学大学院工学府物理情報工学専攻　〒 240-8501 神奈川県横浜市保土ヶ谷区常盤台 79–5
†† 横浜国立大学大学院工学研究院　〒 240-8501 神奈川県横浜市保土ヶ谷区常盤台 79–5
E-mail: †{ishiyama,nakata,kobayashi}@ubicg.ynu.ac.jp, ††kimio@ynu.ac.jp

あらまし　近年，情報システムの信頼性の向上を目指す分野において，ディペンダビリティという概念が注目されている．本研究ではディペンダブルシステムにおける実行時情報の重要性に着目し，さらにオープンプロセスとシステム外部の情報を活用してユーザとシステムとの良いインタラクションを実現する事によるディペンダブルなシステムの構築を目的とする．本稿では上記を達成する手法としてオブジェクトブロギングシステムと dFeed を提案し，ブログを用いて情報システムのカーネルモニタリングの出力であるログ情報を共有するシステムとしてコログを構築，実装した．

キーワード　ブログ，オブジェクト，Linux

Colog: Kernel Monitering Log Sharing System for Dependable System

Naoki ISHIYAMA†, Shinpei NAKATA†, Satoshi KOBAYASHI†, and Kimio KURAMITSU††

† Faculty of Engineering, Yokohama National University　Tokiwadai 79–5, Hodogaya-ku, Yokohama-shi, Kanagawa, 240–8501 Japan
†† ??? Division, Yokohama National University　Tokiwadai 79–5, Hodogaya-ku, Yokohama-shi, Kanagawa, 240–8501 Japan
E-mail: †{ishiyama,nakata,kobayashi}@ubicg.ynu.ac.jp, ††kimio@ynu.ac.jp

Abstract　Recently, the term "Dependability" has become widely known as a one of the degree of system fault tolerance or reliability. We focus on the dependability of running services, which provides continuous service to consumers. Our idea is that the key element of exploiting dependability is in the open community that known as a fundamental community of open source software development. We design and implement both dFeed, and object blogging system to integrate log cooperate system, Colog. Kernel monitoring technique is used as a method to probe or trace deep inside the kernel, which we consider it as a function provider for all services and hold all of critical information for service process.

Key words　blog, object, Linux

1. はじめに

近年，情報システムの信頼性の向上を目指す分野において，ディペンダビリティという概念が注目されている．ディペンダブルという用語はフォールトトレラントとほぼ同じ意味で使用されており，フォールトレラント，またはディペンダブルシステムとは，ユーザにサービスを提供するシステムの故障 (フォールト) は避けられないものとし，そのフォールトによるサービスへの影響を抑えることを目的としたものである．しかし，100%Fault-tolerance な設計，その実装は非常に困難である．また，従来のディペンダブルシステムではシステム運用前に設定したフォールトモデルにしか対応できない為，設計時に想定していなかったフォールトが実行時に発生する場合が問題になる．この様に，情報システムの実行時に起こりうるフォールトを認識する必要が生じている．この実行時のフォールトに関する情報は，情報システムにおいてはオペレーティングシステム，中でもカーネルが把握していると考えられる．これは，カーネルが実行中のサービスの情報，コンピュータ資源の情報を把握していることによるものである．これらカーネルが把握している情報から，潜在的なフォールトに関する情報を取りだす手段として，カーネルモニタリングがあり，一般にロギングシステムとして広く利用されている．ここで我々は情報システムの運用を阻害する情報は，ローカルでのカーネル内部情報だけでなく，ログ情報を共有する事によるコミュニティモデルを

利用したオープンプロセスを導入する事でシステム外部からも得られるのではないか，との仮定の元に本稿で提案するオブジェクトブロギングシステムによる情報システムのディペンダビリティの向上を目指す．

本研究での最終的な目的は，ユーザとシステムとのよいインタラクションを実現する事によるディペンダブルなシステムの構築である．本稿では，Linux 環境において，実行時のロギング情報を動的に指定して取得，集約する事を目的とする．情報システムのロギング情報のブログへの自動投稿システムをオブジェクトブロギングシステムとして提案し，これを用いて実行時のフォールト情報の共有と，それによるディペンダビリティの向上を試みる．

本稿の構成は以下の通りである．第 2 章で本研究の動機について述べ，第 3 章では本研究のシステムデザインについて述べ，第 4 章では本研究で実装したシステムについて述べ，第 5 章では関連する研究について述べる．最後に第 6 章で本稿を総括する．

2. 動　　機

本章では本研究の動機について述べる．

2.1 ディペンダブルシステムにおける実行時情報の重要性

1 章でも述べたように，ディペンダブルシステムは通常，設計時に想定したフォールトモデルしか扱うことができず，実行時に予期しないフォールトが発生した時に対応できない．よって予期しないフォールトに対応するには，実行時にいかに回復を行うかが重要になるが，その前にフォールトの発生箇所を突き止める必要があり，そのためには実行時の情報，しかもサービスにとってクリティカルな情報にアクセスする必要がある．回復させたいサービスはすなわち情報システムにおいてはプロセスであるので，プロセスを管理しているカーネルから情報を取得する必要がある．

2.2 オープンプロセス

Eric S.Raymond による”The Cathedral and the Bazaar”に示されているバザール方式は Linux の成功以降，様々なプロジェクトでその実力が示されてきた [1]．オープンプロセスの特徴としては，コミュニティからのフィードバックがある事，またはパッチなどが頻繁にやり取りされる事や，ソースコードからドキュメントまですべてオープンである事，などがあげられる．このことから，我々はコミュニティにソフトウェアの品質，つまりディペンダビリティを向上させる知識が蓄積されていると考える．本研究はこの蓄積されたコミュニティ知識をいかに実行時のサービスディペンダビリティに活用するかに焦点をあて dFeed を設計・実装した．詳細は後の 3 章に述べる．

2.3 システム外部の情報の活用

2.2 節で述べたとおり，ディペンダビリティを向上させるにはコミュニティ知識を活用することが重要だと述べたが，加えて，我々はシステム内部のカーネルからの情報だけでなく，システムの外部からも実行時のディペンダビリティを向上させる情報を取得できるのではないかと考えた．例えば 2 つの情報システムの構成要素が近い場合，一方のシステムのカーネルモニタリング情報が他方のシステムに対しても有効であると考えた．つまり，カーネル内部のログの情報をシステム間で共有することによって，あるユーザが自分の環境に似た環境で起こったフォールトやエラーの情報を得る事ができるようになる．これによって，システム実行時のフォールト情報がシステムの外部からの取得も可能になり，上で述べたコミュニティモデルとの相乗効果でディペンダビリティの向上を目指せるのではないかと考える．

3. システムデザイン

本章では，2 章で述べた動機に基づいて本稿で提案するシステムのデザインについて述べる．本稿で提案するシステムは大きく 2 つに分かれる．まず，カーネル内の情報の共有化をはかるオブジェクトブロギングシステムについて述べ，次にカーネルモニタリングの遠隔操作の為の仕組みとして dFeed について述べる．

3.1 オブジェクトブロギングシステム

我々は 2.3 節で述べたように，カーネル内部の情報を共有することにより，ローカルな情報システムの内部情報だけでなく，外部の情報システムからフォールト情報を得られるようになる為，ディペンダビリティを向上させる助けになると考えている．そこで，本稿ではカーネルモニタリング情報の共有の自動化を目的とする，オブジェクトブロギングシステムを提案する．オブジェクトブロギングにおけるオブジェクトとは，カーネルや各種デーモン，ユーティリティーなどログを生成する主体を指す．通常，ログにはシステム上に起きた出来事 (イベント) が反映されているので，これにはフォールト情報が含まれていると考えられる．オブジェクトブロギングでは，オブジェクトからイベントを収集し，それをブログに 1 つの記事 (エントリ) として自動的に投稿する．これによりフォールト情報を含むログ情報の共有の自動化を図る．オブジェクトブロギングの全体図を図 1 に示す．

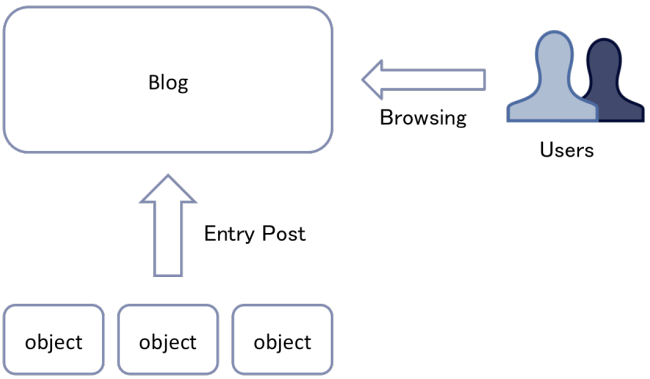

図 1　オブジェクトブロギングシステム概要

オブジェクトブロギングシステムの特徴として，人ではなくモノ (オブジェクト) がブログに書き込む事があげられる．これにより自動的にログが Web 上で共有され，フォールト情報の共有化が可能となる．しかし，ローカル情報のログの量は膨大であり，これを全てブログに書き込むことは現実的ではなく，ロ

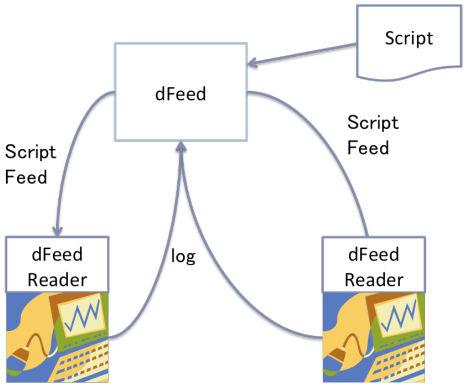

図 2　dFeed 概要

グ情報のフィルタリングが重要となる．本稿ではこれを dFeed という機構で解決した．dFeed については次節で詳しく述べる．

3.2　dFeed

dFeed はログを取る対象をスクリプトで記述し，ローカルマシン内でのフィードリーダによってスクリプトをダウンロードすることによって，ロギングのフィルタリングを行う機構である．この機構により必要に応じてカーネルモニタリングによるロギングを制御することが可能となる．dFeed の概要を図 2 に示す．まず dFeed にカーネルモニタリング制御用のスクリプトが渡されて登録される．次に各ローカルマシンの dFeedReader がスクリプトの更新を確認しにいく．ここで更新があれば，ユーザの判断によりスクリプトがダウンロードされて実行され，各マシンのログが dFeed に送信される仕組みとなっている．dFeed を用いる事によって，スクリプトの開発者へのフィードバックが期待できる．

4.　実　　装

本稿では 3 章で提案したオブジェクトブロギングを応用したログの共有システムとしてコログを構築，実装した．なお，クライアント側である各ローカルマシンは Linux 環境を想定しており，カーネルモニタリングのツールとして systemtap を採用した．コログシステム全体の概要を図 3 に示す．

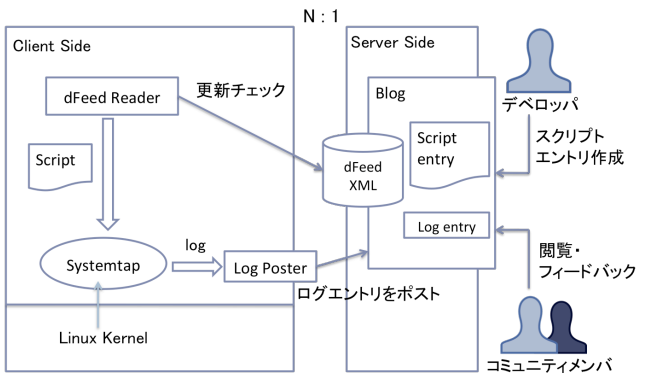

図 3　コログシステム全体概要

コログシステムのブログ部分には 2 つの記事 (エントリ) の種類がある．1 つはカーネルモニタリングの挙動を制御するスクリプトを表現するスクリプトエントリであり，これは主に systemtap へ渡すスクリプト本体と，そのスクリプトの説明，スクリプトを作成したユーザ名からなる．もう 1 つのエントリはローカルマシンから自動的にポストされるログを表わすログエントリである．ログエントリは主に送信元であるマシンのホスト名と，そのログの内容からなる．ブログの TrackBack 機能を利用して，ログエントリからそのログを制御したスクリプトエントリへ TrackBack を打つことにより，ブログ内でのスクリプトに関連するログの一覧性を高めている．また，スクリプトエントリの TrackBack 数でそのスクリプトを評価する事も可能であると考える．

次に，コログでのカーネルモニタリング操作の処理の流れについて述べる．まずスクリプトをデベロッパが blog に登録すると，dFeed 用の XML が更新される．各ローカルマシンの dFeedReader はこの XML をチェックする事によってスクリプトの更新を確認する．ここで，ユーザがスクリプトの実行を承認した場合，スクリプトは systemtap に送られて実行される．systemtap からのログ情報を LogPostmaster が受けとり，blog にログエントリとして書きこむ．このときログのエントリはスクリプトエントリに対して TrackBack を打つ．コミュニティメンバはこの blog を閲覧する事によってフォールト情報を得る事が可能となる．

5.　関 連 研 究

情報の提示にブログを用いた関連研究として，前川氏らの Object-Blog System for Environment-Generated Content [2] が挙げられる．[2] ではユビキタスセンサ (RFID 等) を用いて家具やドアといったモノから現実の環境で起きた現象の情報を収集し，その情報を基に生成されたコンテンツを環境生成コンテンツ (EGC) としている．この EGC を提供するサービスとして Object-Blog System を構築している．システムなどの情報をブログで閲覧するという点では一致するが，本研究ではブログの投稿を利用して，カーネルをモニタリングするという点で大きく異なる．

また，このようなカーネルモニタリングを可能にするシステムとしては，本研究でも利用した systemtap や dtrace [4] といったものがある．しかし，これらは専用のスクリプト言語を用いてカーネルモニタリングの結果を標準出力に出力する．しかし出力先は標準出力のみなうえ，リモートでのカーネルモニタリングの制御はもちろんできない．他にも TOSKANA [5] や KLASY [6] など様々なカーネルモニタリングシステムが存在するが，カーネルのトレースの性能を向上させたものが殆どで，Web 上に出力したり，リモートでカーネルモニタリングを実行可能にするものではない．

カーネルのログをとりリモートでも扱えるものとして，syslog がある．しかし syslog のログはあらかじめログを取るようにプログラムしなければならなく，カーネルをトレースすることはできず，ただ初めの指示通りログを吐くだけである．さらにはリモートといっても情報を得るだけで，こちらからカーネルモニタリングの指示を出すことはできない．

このように，カーネルモニタリングをブログのような web コミュニティツールを利用して実現した例は本研究が初めてだと考えられる．

6. ま と め

本稿ではディペンダビリティを向上させる為にカーネルモニタリングの出力であるログ情報の共有化を目的とし，それを実現するシステムとしてオブジェクトブロギングシステムと dFeed を提案，クライアント環境として Linux を対象とし，コログシステムの構築と実装を行なった．

今後の課題としてカーネルモニタリングの結果，エラーが発見された場合にコログシステムを用いてこのエラーを是正したという要求を満たすことが考えられる．

文 献

[1] Eric S.Raymond, The Cathedral and the Bazaar, http://www.catb.org/~esr/writings/cathedral-bazaar/ .

[2] T. Maekawa, Y. Yanagisawa, Y. Kishino, K. Kamei, Y. Sakurai, and T. Okadome: Object-Blog System for Environment-Generated Content, IEEE Pervasive Computing, Vol. 7, No. 4, pp. 20-27 (Oct.-Dec. 2008).

[3] V. Prasad, W. Cohen, F. Ch. Eigler, M. Hunt, J. Keniston and B. Chen: Locating System Problems Using Dynamic Instrumentation, Linux Symposium, 2005

[4] C. Bryan M., S.Michael W. and L. Adam H.: Dynamic instrumentation of production systems, ATEC '04: Proceedings of the annual conference on USENIX Annual Technical Conference, 2004

[5] E. Michael and F. Bernd: Supporting autonomic computing functionality via dynamic operating system kernel aspects, AOSD '05: Proceedings of the 4th international conference on Aspect-oriented software development, pp. 51-62, 2005

[6] Y. Yoshisato and K. Kenichi and C. Shigeru: A dynamic aspect-oriented system for OS kernels, GPCE '06: Proceedings of the 5th international conference on Generative programming and component engineering, pp. 69-78, 2006