Vol.2009-DPS-140 No.13
2009/9/11

ログに基づくライセンス管理方式の提案

友野 敬大[†]　上原 稔[†]　島田 裕次[†]

近年，米国企業に留まらず，日本企業でも不祥事が多発し，内部統制の必要性が急速に高まっている．中でも，ログ監査は重要であり，これが適切に行われていない内部統制システムは，脆弱なものになる．また，ログには最終防衛的な意味に加えて企業改善につながる情報が多く含まれている．したがって，ログの的確な活用は，企業において有益である．しかし，ログ監査のためのシステム導入や管理・運用のコストは決して安価ではなく，ログのデータ量に比例してコストがかかる．また，今日では，企業や学校などの組織では，ソフトウェアを購入したにも関わらず，利用されていない場合が少なからずある．高価なソフトウェアのライセンスが正しく利用されていない場合には，無駄なコストとなってしまう．我々は，VLSD を用いた低コストかつ半永久的にログを保存可能なシステムを開発した．本研究では，このシステムとデータベースを連携することによって，ログからソフトウェアの利用率を導出し，有効にライセンスが活用されているか確認し，適切に管理する方法を提案する．

A Proposal for Method to Manage License based on Log

AKIHIRO TOMONO[†]　MINORU UEHARA[†]
YUJI SHIMADA[†]

In recent years, many accounting scandals have been reported in companies not only in the United States, but also in Japan. The need for Internal Control is growing steadily. In particular, auditing logs is important for Internal Control. Internal Control lacking audit is incomplete. In addition, logs are included not only defensive implication but also a lot of information to lead to improve company. And thus the precise make use of the log can be beneficial in a company. However, the cost of an information system is dependent on the amount of data, which in the case of log data can be very large. Today, there is not a little the case that is not used software though they purchased it in the organizations such as a company or the school. When a license is not used definitely, it is a waste cost. We have overcome the capacity problem by developing the Virtual Large-Scale Disk (VLSD) toolkit, which allows us to construct large-scale disks. Using this toolkit, we have implemented log storage with low cost and a long lifetime. In this paper, by using this system with a database, we suggest a method to manage confirming whether a license is used effectively giving the using rate of the software from logs.

1. はじめに

近年，米国企業に留まらず，日本企業でも不祥事が多発し，内部統制の必要性が急速に高まっている．内部統制とは，COSO(Committee of Sponsoring Organizations of the Treadway Commission)によれば，『(1) 業務の有効性・効率性，(2) 財務報告の信頼性，(3) 関連する法規の遵守の 3 つの目的の達成に関して，合理的な保証を提供することを意図した，会社の取締役会，経営者およびその他の従業員によって遂行されるプロセスであり，相互に関連する要素，すなわち，統制環境，リスクの評価，統制活動，情報と伝達，モニタリングから構成される』と定義されている．本論では，これを実現するための一連の仕組みを内部統制システムという．

日本 IBM やサン・マイクロシステムズなどのような大手企業では，独自の内部統制システムを構築しているケースが多々ある．システム構築を外注すると，大企業であればあるほどシステム規模が大きくなり，その分のコストが発生することになる．

企業では，コンプライアンス確保のために，内部統制を実現するためのツールを選択し，活用する必要がある．しかし，現在の普及品はどれも高価であり，導入コストだけではなくデータ量に応じた運用コストについても考慮しなければならない．実際のログ（以下，生ログとする）は，何も加工されていないため，その証拠能力は比較的高いものと考えられるが，ログはシステムの規模に比例して，確実に保存領域を圧迫していく．我々は OS の機能を利用し，半永久的にログを保存可能なシステムを開発した．VLSD を用いて容量を確保するので，NAS や SAN などといったストレージシステムを導入しなくても，低コストで大容量のストレージを利用できる．

ログは，システムの最終防衛的な意味合いをもつとともに，さらに企業改善につながる情報を多分に含んでいると考えられる．これらの情報は，的確に利用すれば企業に利益をもたらす．また，業務上不必要なソフトウェアやユーザの操作を把握することによって，内部統制を実現する上でのリスク回避にも対応できる．

一方，今日では，企業や学校などの組織において，ソフトウェアを購入したにも関わらず，利用されていない場合が少なからずある．高価なソフトウェアのライセンスが正しく利用されていない場合，それは無駄なコストと言える．企業改善をする上で，遊休ライセンスを起因とする無駄なコストは削減すべきである．また，インターネットを通じて不正入手したソフトウェアや違法コピーなど，ソフトウェアの管理責任についても強く問われている．そこで，ログからソフトウェアの利用率を導出し，有効にライセンスが活用されているか，不正なソフトウェアが利用され

[†] 東洋大学
Toyo University

ていないかどうかを確認し，適切に管理する方法についても提案する．

ログは，一度データベースに格納し，その上でログの中から必要な情報を抽出する．生ログは VLSD を用いて半永久的に保存し，必要に応じデータベースに格納して扱うことで，効率的な管理方式を実現する．

2 章で本研究の関連研究，3 章でシステムの提案，4 章でシステムの考察，5 章で今後の課題について言及し，6 章でまとめる．

2. 関連研究

ここでは本研究の関連研究として，本研究で提案するシステムの実装に必要な VLSD とそのセキュリティ，ログ管理システムについて述べる．

2.1 VLSD (Virtual Large Scale Disk)

通常，大規模ストレージを構築する際，最初に検討されるのは分散ファイルシステムで，空き容量をマウントする方法だろう．しかしながら，NFS を始めとするファイルシステムレベルの分散ストレージでは，空き容量を連結して 1 つのストレージにすることはできない．ディスクレベルの分散ストレージが必要になる．

VLSD[1) 2)]とは, Java によるソフトウェア RAID と NBD の実装を含む大規模ストレージ構築のためのツールキットである．VLSD は 100% pure Java であり，Java が動作するプラットフォームの上なら VLSD も動作する．そのため Windows や Linux が混在する環境に適している．VLSD を用いると OS に制約されることなく NBD デバイスと RAID を自由に組み合わせることができる．最低限必要な NBD デバイスはファイルサーバの 1 つである．

PC 教室や会社の遊休資源（HDD の空き容量）を連携し，1 つの大きな仮想ストレージを構築して用いる．VLSD ツールキットは多様なクラスを組み合わせることによって，8EB までの大容量ストレージを実現する．これを用いて，512 台の PC で構成される PC 教室のために，70TB のストレージの試作に成功した．ただし，ストレージは大きくなるほど，信頼性が低下するため RAID を用いて信頼性を上げる必要がある．ここでは，512 台のディスク（1 ディスク=170GB）を 32 グループにして RAID66 を構築する(図 1 に示す).

また，NBD を用いることで，ファイルシステムに依存しないディスクレベル分散ストレージの実現が可能となる．これにより，高価なストレージの必要性がなくなる．このような分散型ストレージは，HDD 代のみとなるので集中型の費用と比較すると, そのコストを大幅にカットすることができる. VLSD の構成図を図 2 に示す.

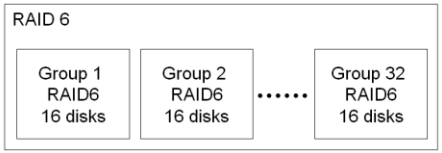

図 1　RAID66 の構成図
Figure 1 Composition of RAID66

図 2 に示すように, 試作システムは 64 ビットファイルサーバとディスクサーバがある．ディスクサーバの Linux や Windows などの OS からなる仮想ディスクは，ディスクの読み込みや書き込みを Java の RMI で機能を提供する．ファイルサーバの方は用意されたディスクに接続して RAID66 を構築する．

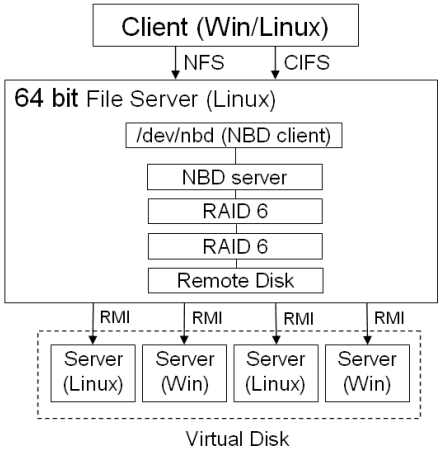

図 2　VLSD の構成図
Figure 2 Composition of VLSD

RAID66 とは, 2 階層の RAID6 である. NBD Server はその RAID66 を利用して NBD Client からのアクセスを待つ．そして，NBD Client の起動をした後，XFS でフォーマットする．Windows のクライアントは Samba を介してそのファイルサーバをアクセスする．また Linux の場合は NFS を用いて接続する．

NBD プロトコルはセキュリティ上脆弱なので, ネットワーク上で運用するのは危険である．しかし，我々の方式では 1 台のサーバ内のプロセス間通信として NBD

を用いているため安全に運用できる．実際の C/S 間通信はセキュリティを考慮した RMI に基づくプロトコルで実現される．

2.2 VLSD のセキュリティ

ログの保存容量の問題を解決する VLSD には，実際の運用において，内部統制を実現するために，そのセキュリティを確保することが問題となる [3]．具体的には以下のような問題が挙げられる．

・PC における盗聴および改ざん
・通信路における盗聴および改ざん
・なりすまし

これらの問題を解決するセキュリティ技法について考える必要がある．VLSD のセキュリティは，AAA アーキテクチャに基づく．それぞれ，認証（Authentication），認可（Authorization），監査（Audit）の頭文字を取ったものである．

VLSD による大規模ストレージは，多くのクライアントマシンの容量の一部を収集して構築されている．そのため，サーバ OS がアクセス制限をかけても，クライアント上から直接不正なアクセスを許してしまう．その原因として，リモートディスクへのアクセスを許すことと，その内容を読み取れてしまうことが挙げられる．アクセス権の問題は，ディスクを提供している以上，管理者である可能性は大いにあるので，解決するのは困難である．しかし，後者の原因を解決すれば，内容は読み取られないので，データの保護をすることができる．その方法としてディスクの暗号化を挙げる．VLSD は，暗号化を実装するクラスを持つ．暗号化することで，クライアントマシンからは内容を読み取ることができなくなる．

また，改ざんを防ぐ方法として，書き込みを禁止する方法が挙げられる．書き込みが必要な場合には，別のディスクとして書き込み，その差分を検証すればよい．VLSD は ReadOnlyDisk というラッパークラスを持つ．これをラップすることによって書き込み禁止ディスクを実現できる．

暗号化のオーバーヘッドは，通常のディスクアクセスなら無視できないものだが，ログを参照する機会はそう多くないと考え，高速性よりも暗号化による安全性を選択すべきだろう．

2.3 ログ管理システム

ログ管理にはいくつかのレイヤがある．生ログを収集・保存するレイヤ，データベースを用いて管理するレイヤ，そこから情報を取り出し，グラフ化や関連付けをするレイヤである．図 3 にログ管理のレイヤを示す．

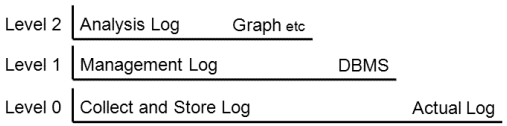

図 3 ログ管理のレイヤ
Figure 3 Layers of Log Management

今日では，ログをデータベース化して管理および解析するのが通例であるが，このシステム [4]では，何も手を加えてない生ログに着目し，保存・管理することによってログの信頼性をより高められると考える．システムが排出したそのままの生ログは，証拠能力が最も高いものだと考えられるからである．データベース化することは，ユーザの使いやすさの向上につながるが，その処理を行うことで何らかの情報が損失する可能性も大いに考えられる．また，厳密に言えば，データベース化はログの加工であり，その間に不正な加工や欠損がないとは言い切れない．

今日ではログをデータベース化することが多く，またそのような製品も多く提供されている．逆に言えば，生ログを保存・管理するシステムはほとんど普及しておらず, また通常はそのような管理方法を採用しないだろう. その大きな理由として，容量の問題が挙げられる．単体としては大きくなりにくい（と考えられがちな）ログファイルだが，収集していけば確実にディスクを圧迫することになる．そこで，我々は VLSD を開発し，それを用いて生ログの保存・管理を行う．システム構成図を図 4 に示す．コンピュータの遊休資源を用いるので，新たな設備を追加する必要はない．そして，将来的には生ログを保存した上で，随時 DBMS と連携する．具体的には，Level0 と Level1 の間にログの正規化を行う必要がある．これにより，機能性とログの証拠性を両立できると考える．

このシステムでは，OS にあらかじめ備わっている Syslog や Logrotate といった機能を利用してシステムを開発する. 従来では, Syslog が排出したログは上書きされ，古いログは参照できなかった．Logrotate のデフォルトの設定が月に 4 回ログをローテーションし，古いログを上書きしていくため，最高でも 4 週間前のログしか残らないことになる．ローテーションする回数を増やせば，ある程度は過去のログを参照できるが，それよりも古いログをニアラインストレージにコピーして管理する方が有益である．ニアラインストレージは，本大学の Web サーバやファイルサーバなどの常時稼働しているサーバ 25 台から遊休資源を収集して，構築する．その構成を図 5 に示す．

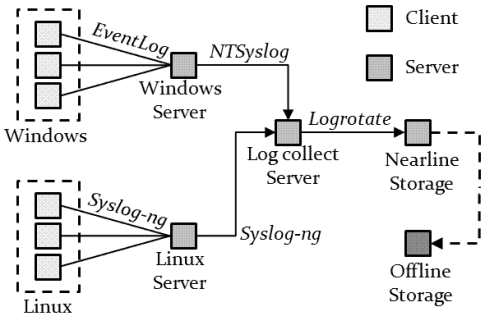

図 4　システム構成図
Figure 4 A Log Management System

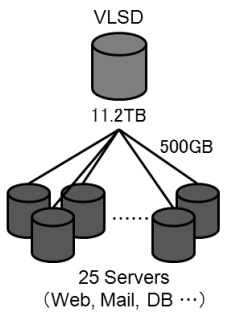

図 5　サーバファーム
Figure 5 Sever Farm

3. ログに基づくライセンス管理システム

3.1 システム概要

ライセンス管理方法として，ユーザが業務時間中にどの程度，どのようなアプリケーションを使用したかという観点が挙げられる．すなわち，アプリケーションを利用した時間を求めれば，実際に必要なものなのかを判断する材料になり得る．例えば，ある一定期間使用されてないアプリケーションのライセンスはもちろん無駄と言えるが，利用時間がごくわずかなものに関して言えば，操作するユーザおよびインストールするマシンを限定し，統一することで遊休ライセンスを解消できる．

また，自動的にログを収集するエージェントをクライアントマシンに配置することで，ユーザに余計な負担をかけずにログを収集する．自動収集エージェント（以降，エージェントとする）は，業務に関係のないアプリケーションの使用や不審な操作も検出する．これにより，組織におけるインストールされているアプリケーションの状況を把握することで，関係する法的なリスクを回避することができる．

ユーザがアプリケーションを利用したかどうかを判断するのに，そのアプリケーションがアクティブになっている時間を計測して用いる．本システムの構成図を図 6 に示す．それぞれのアプリケーションに対してこれを行い，終業時間などの 1 つの区切りに，計測した時間をそれぞれ合計する．合計したものを降順にソートして表示することで，その日のユーザが使用したアプリケーションの中で利用時間が多いものを把握できる．

本研究においては，クライアントマシンにエージェントを配置してログの収集を行うため，悪意あるユーザがプログラムを終了すること，及びログが改ざんされて

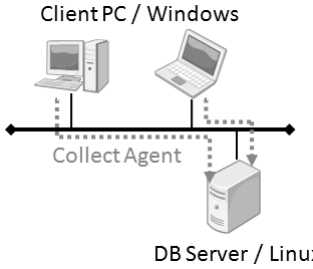

図 6　クライアントからサーバへの流れ
Figure 6 A Flow of C/S

いない場合を想定する．また，エージェントは Windows の API フックによって実現されるため，Windows 以外の OS での動作は保障しない．このエージェントは VB.NET で記述されているため，.NET Framework の環境が必須である．

3.2 自動収集エージェント

まず，エージェントは，アプリケーションがアクティブになったときに，そのウィンドウハンドルを取得する．さらに，取得したウィンドウハンドルから，そのプロセス ID やプロセス名などの情報を取得する．エージェントは，そのアプリケーションがアクティブである間，ウィンドウハンドルを保持し続け，時間を測定する．そして，他のウィンドウハンドルがアクティブになったときにログを書き出す．書き込む際，日付，ユーザ名及びホスト名などを取得する．書き出されるログの書式は次のとおりである．また，利用時間は，秒単位で計測される．

“日付”,”プロセス ID”,”プロセス名”,”ユーザ名”,”ホスト名”,”アプリケーション名”,”利用時間”

プロセス名とアプリケーション名は必ずしも同じではない．プロセス名とは，アプリケーションのウィンドウを生成したプロセスのプロダクト名である．例えば，電卓は OS によって提供される．そのため，プロセス名は”Microsoft(R) Windows(R) Operating System”であり，アプリケーション名は”電卓”となる．また，Web ブラウザでは，プロセス名が”Windows(R) Internet Explorer”であるが，アプリケーション名には，閲覧している Web サイトのタイトルが表示される．すなわち，アプリケーション名のほうが，詳細な情報を含んでいる場合が多い．

書き出されたログを VLSD に保存することで，半永久的に保存が可能となる．こ

のログをデータベースサーバに転送し，格納する．

3.3 データベース

エージェントによって書き出されたログをデータベースに格納し，管理および活用する．データベースには MySQL を用いる．今回使用したバージョンは 5.0.45 である．格納されたログの情報に対して検索を行い，プロセス ID が一致するものの利用時間の和をとる．

データベースには日付を名前としてテーブルを作成し，そこにユーザあるいはホストごとにエージェントから書き出されたログを格納する．プロセス ID は一意な値だが，異なるホスト間では一意とは限らない．これは，ユーザごとに合計を出すことで解決する．また，これにより組織内において何人のユーザがアプリケーションを利用したのかが分かる．

ログは VLSD で半永久的に保存するが，データベースは必要になったとき毎に構築する．ログが保存されてさえいれば，過去の情報までデータベース化して参照することが可能だが，過去に使用したデータベースは利用価値や頻度が低下すると考えるからである．

データベースの構造を図 7 に示す．

Field	Type	Null	Key	Default	Extra
lid	int(10) unsigned	NO	PRI	NULL	auto_increment
date	char(32)	NO			
ps	varchar(32)	NO			
psn	mediumtext	NO			
usr	char(64)	NO			
host	char(64)	NO			
app	mediumtext	NO			
time	int(10) unsigned	NO			

図 7　データベースの構造
Figure 7 Structure of Database

4. 考察

アクティブなウィンドウが切り替わるたびにログが出力されるため，その量は膨大なものになると予想される．実際に運用した場合を考える．エージェントを起動したまま作業をした結果，ホスト 1 台あたり 8 時間でおよそ 300KB，ログが 1,300 行となった．これをホスト 512 台としたとき，ログのサイズが 154MB 必要になる．行数はおよそ 67 万行となり，当然データベースで処理する行数もそれに比例して多くなる．そのため，適切なログ保存用のストレージを構築する必要がある．前述の通り，VLSD を用いてこの問題を解決する．サーバから遊休資源を確保し，ストレージを構築するが，その規模についても再考しなければならない．関連研究で挙げたログ管理システムでは，本大学で常時稼働しているサーバ 25 台から遊休資源を確保することを考えている．実際には，中小企業において常時稼働しているサーバ台数はそこまで多くはない．仮に，常時稼働しているサーバが 5 台だとする．それぞれ 500GB ずつ遊休資源を収集して，RAID6 でストレージを構築する．これにより 1.5TB の要領を確保できる．ログの必要領域から算出すると，充分である．

さらに，ライセンス料の低減について考える．我々は，研究用に VMware Workstation を導入している．1 ライセンスあたりおよそ 2.5 万円なので，遊休ライセンスが数本あれば 10 数万円近くのコストを無駄にしていることになる．研究室だけではなく，企業においても同様なことが言える．高価なアプリケーションであればあるほど，早急に対応が必要である．また，大学におけるキャンパスアグリーメントでは大学全体で一括してライセンス契約をするので，リプレイスなどの際にライセンス数をその都度見直さなければ，利用していないライセンスについても支払が発生しさらに大きな単位の無駄を生じることになる．本システムでは，API フックによって得られる情報を利用することによって，そのコストは大きく削減できると考える．

実際に書き出されたログの一部を図 8 に示す．

lid	ps	usr	psn	SUM(time)
15	<3256>	uehara_minoru	VMware Workstation	4530
16	<2732>	uehara_minoru	2007 Microsoft Office system	2873
61	<6048>	uehara_minoru	Windows(R) Internet Explorer	1499
18	<-1>	uehara_minoru	0	1070
2	<4076>	uehara_minoru	Microsoft(R) Windows(R) Operating System	991
432	<6036>	uehara_minoru	Google Chrome	295
421	<1788>	uehara_minoru	Tera Term Pro	44
4	<4476>	uehara_minoru	APPGet	35
1	<3052>	uehara_minoru	Sleipnir	21
294	<2492>	uehara_minoru	Windows Internet Explorer	18
114	<456>	uehara_minoru	Winamp	3
196	<5332>	uehara_minoru	Trend Micro Internet Security	3
483	<2784>	Yasu	Microsoft Office 2003	9499
757	<2744>	Yasu		2709
471	<1328>	Yasu	Microsoft(R) Windows(R) Operating System	474
472	<-1>	Yasu	0	184
534	<3964>	Yasu	Trend Micro Internet Security	2
768	<460>	Yasu	BOINC client	2

図 8　収集したログの一部
Figure 8 A Part of the Collected Log

図 8 中に示されているプロセス ID が”-1”になっているものは，何もアクティブではない状態のときに見られる．すなわち，スクリーンセーバやユーザロックである時間を測定している．このとき，プロセス名は取得できないので，意図的に”0”を値として代入している．また，ユーザ Yasu において，プロセス名が取得できていないアプリケーションがある．生ログを調査した結果，これは Eclipse であることがわかった．Eclipse とは，IBM によって開発された統合開発環境(IDE)の 1 つである．このように，アプリケーション側でプロダクト名が設定されていない場合は，その値は NULL となり，参照することができない．この場合は，図 9 のようにプロセス ID からアプリケーション名を得ることができる．

```
+-----+--------+-----------------------------
| lid | ps     | app
+-----+--------+-----------------------------
|   1 | <3052> | Sleipnir - [vb.net プロセス
|   2 | <4076> | Release
|   4 | <4476> | APPGet
|  15 | <3256> | CentOS 5.3 - VMware Workstati
|  16 | <2732> | DPS140_1st.docx - Microsoft W
|  18 | <-1>   | <0>
|  61 | <6048> | ToyoNet Hacks: Microsoftキャ
| 114 | <456>  | 1.Vinyl and Circuitry July 24
| 196 | <5332> | ウイルスバスター2007
| 294 | <2492> | MSN Japan - Windows Internet
| 421 | <1788> | Tera Term
| 432 | <6036> | 無題 - Google Chrome
| 483 | <2784> | Microsoft PowerPoint - [中間
| 757 | <2744> | Eclipse
| 768 | <460>  | World Community Grid
+-----+--------+-----------------------------
```

図 9　PID からのアプリ名参照
Figure 9 Refer to App name from PID

5. 今後の課題

本システムの実用性をより高めるためには，以下のような課題を解決する必要がある．

(1) ログ収集の見直し

前述の通り，エージェントによって取得できない情報がいくつかあることが判明した．エージェントは，FileVersionInfo の ProductName をプロセス名として参照している．FileVersionInfo が保持する情報にはこれ以外にも，内部名(InternalName)やオリジナルファイル名(OriginalFilename)など，判別に利用できる可能性があるものが多くある．必要に応じて，組み合わせて用いることも考えられる．この場合，保存容量は多く要求されるが，あらゆる情報を記録しているログを生成すれば，様々な状況に対処できる．そのため，記録するフィールドを再考し，ログの書式を検討する必要があると考える．

(2) データベースとの自動同期

現段階では，データベースは必要なときに構築する仕様になっている．テキストベースで記録されたログをファイル毎に渡す．この方式のメリットは，作成したものの,そのデータベースをアクセスしなかった,というような状況がなくなるので，必要な保存領域を低減できる．しかし，いざというときに構築してから参照しなければならないので，管理者には若干負担があると言える．エージェントは 1 つの区切りでログを書き出すので，それをトリガとしてデータベースを構築する方法が考えられる．必要保存領域は多くなるが，管理者の負担を減らすことができる．リアルタイムでログとデータベースを同期させる必要があるのかについて検討する必要があると考える．

(3) 結果の視覚化

図 8 で示すように，ソートされているものの，結果は数値によるものなので，伝わりにくいケースが考えられる．直感的に理解するには，視覚化が一番だろう．前述のようにログ管理を 3 つのレイヤに分けると,視覚化は Level2 に該当する.Level2 は証拠性より，使いやすさを優先する．特に，1 日の利用時間の割合は，グラフ化することで直感的に理解を深めることができると考える．

6. まとめ

本論文では，ログからクライアントのアプリケーション利用状況を読み出し，ライセンス管理に役立てる提案をした．企業などの組織において，ライセンス違反や不正入手および違法コピーが発覚した場合，法的な処罰を受けるだけでなく，社会的信頼は大きく損なわれるのは言うまでもない．ライセンス管理に留まらず，クライアント PC の状況を把握するのは，これから，より一層重要視されると考えられる．

謝辞

本研究は科研費基盤(C)「PC グリッドによる高信頼・高効率な分散仮想ストレージの研究(19500066)」により援助されています．

参考文献

1)上原 稔 “教育環境における仮想大規模ストレージのためのツールキット”, マルチメディア通信と分散処理ワークショップ, pp.205-210, 2006 年 11 月
2)チャイ エリアント, 上原 稔, 森 秀樹 ”PC 教室のための仮想的大規模ストレージの構築”, マルチメディア, 分散, 協調とモバイル（DICOMO2007）シンポジウム論文集, pp.617-622, 2007 年 7 月
3)上原 稔 “仮想大規模ストレージにおけるセキュリティ”, 情報処理学会研究報告書, pp.61-66, 2007 年 11 月
4)Akihiro.T, Minoru.U, : ”A Log Management System for Internal Control”, NBiS2009, (to be appeared)