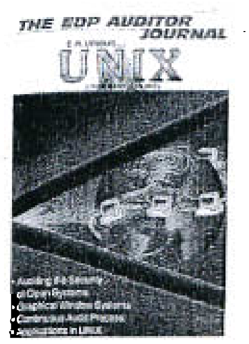


# The Continuous Process Audit System: A UNIX-Based Auditing Tool

*by Miklos A. Vasarhelyi, Ph.D., Fern B. Halper, Ph.D. and Kazuo J. Ezawa, Ph.D.*

## Preface

This paper examines the **Continuous Process Auditing System** (CPAS), a UNIX-based auditing tool developed at AT&T Bell Laboratories for the Internal Audit organization. The system is an implementation of a **Continuous Process Audit Methodology** (CPAM) and is designed to deal with the problems of auditing large paperless database systems.

CPAS was built using the UNIX operating system and the NeWS windowing system. It was constructed using standard UNIX platform tools and enriched by a commercial relational database. This application illustrates that the UNIX system is a rich and effective environment for advanced applications that integrate mainframes and workstations, VMS, DOS and UNIX.

The authors wish to thank the participants of research seminars at several universities and the attendees of the EDPAA, IIA, and AICPA professional meetings for their comments and suggestions. We would also like to thank Mr. Stanley Halper, of the Audit Committee Support Network, for his insightful comments.

## 1. Introduction

Since the introduction of computers in business, the auditor's role has changed, along with the complexity of tasks they need to perform.[1][2][3][4] These changes have created major challenges in performing the auditing and attestation function. For example, 1) the advent of time-sharing and data communications has allowed continuous access to data from many locations, creating access exposures; and 2) database systems have added more complexity due to the lack of obvious mapping between the physical and logical organization of data. Today, many large application systems consist of multiple modules, often with multiple copies of databases, at many locations. The "snapshot" audit, that uses only a few days of data from the system being audited, is not effective in these real time systems because evaluating the controls over these systems requires evaluating the controls at many points in time, which is virtually impossible after the fact, even if a detailed paper trail exists, using this method.

Auditors have developed specialized audit software to deal with traditional audit functions, and have begun to use advanced technologies in support of auditing. For example, Cash et al.[5] examines techniques that can be used to audit Accounting Information Systems. Other examples of these technologies are the use of advanced workstations[6] and decision support systems[7] that incorporate analytic tools and expertise[8] to be used on top of the corporate information system. This paper describes the Continuous Process Audit System (**CPAS**), a UNIX-based system that monitors large complex systems from an audit perspective. The purpose of CPAS is to provide auditors with an integrated diagnostic view of a system. CPAS works by gathering and collating diagnostic data produced from different parts of a system and presenting key checkpoints and analytics in a workstation environment. The purpose of the analytics is to ensure the financial integrity of the system and call auditors' attention to any anomalies. CPAS is an audit application of a general monitoring methodology developed by AT&T Bell Laboratories for the AT&T Internal Audit Organization. The methodology is supported by a set of tools, also developed at Bell Laboratories. This paper focuses on the system aspects of CPAS, and in particular, the UNIX-based features of this implementation. For a more detailed view of the motivation and theory behind CPAS see Vasarhelyi and Halper (1991).[9]

The paper is divided into six sections. In the next section, we provide an overview of CPAS. Section 3 provides an overview of the methodology. Sections 4 and 5 describe how CPAS was implemented. The Discussion (section 6) touches briefly on audit issues surrounding the use and development of CPAS, our evaluation of UNIX for the task of building the CPAS system, and some ideas for future work.

## 2. Overview of Approach

In Continuous Process Auditing, data flowing through a system are monitored and analyzed continuously (i.e., daily) using a set of auditor defined rules. System alarms and reports call the auditor's attention to any deterioration or anomalies in the system. Continuous Process Auditing, then, is really an analytical review technique[10] since constantly analyzing a system allows the auditor to improve the focus and scope of the audit. Furthermore, it is also often related to controls as it can be considered as a meta form of control (audit by exception) and can also be used in monitoring control (compliance) either directly, by looking for electronic signature, or indirectly by scanning for the occurrence of certain events.

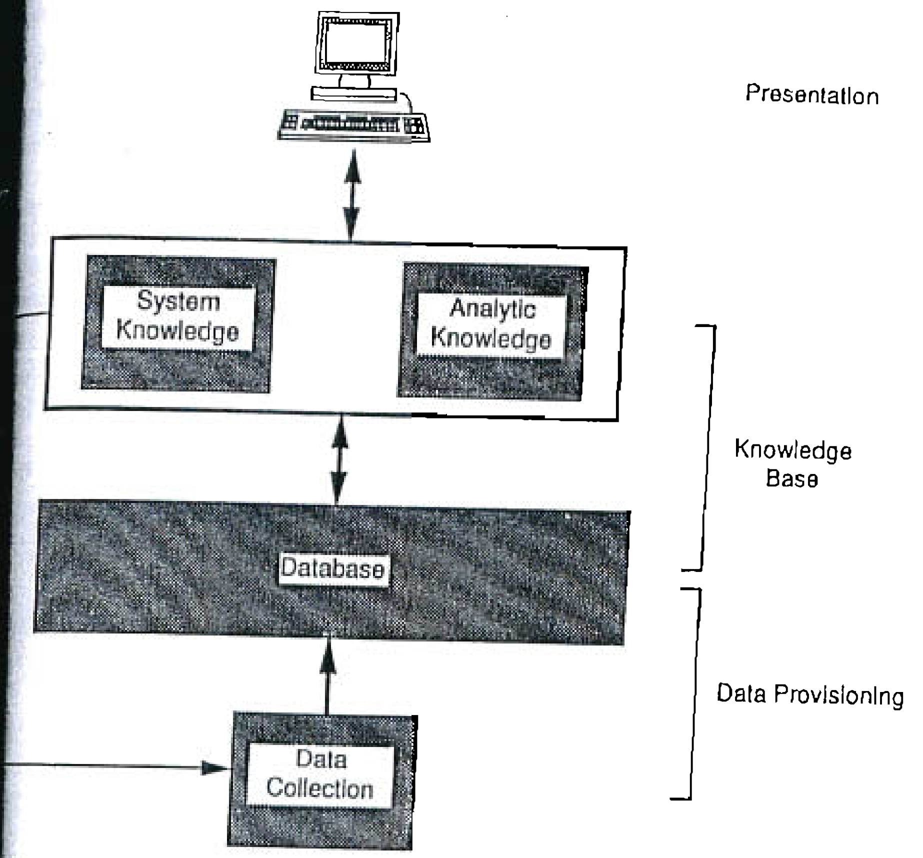
Ultimately, if a system is monitored over time using a set of ever-improving auditor heuristics, the audit can rely purely on exception reporting. Impounding auditor knowledge into the system means that tests that would normally be performed once a year are repeated with each cycle performed or at planned checkpoints.

The adoption of a Continuous Process Audit Methodology (CPAM) will change the nature of evidence, timing, procedures and effort involved in audit work.

- **Evidence:** The auditor will place an increased level of reliance on the evaluation of flow data (while accounting operations are being performed) instead of evidence from related activities (e.g., preparedness audits).
- **Timing:** Audit work would be focused on **audit by exception** with the system gathering knowledge exceptions on a continuous basis.
- **Procedures:** Traditional auditing involves the examination of archival data, substantially after the event and emphasizes paper-based evidence. Continuous Process Auditing involves the examination of archival and immediate data, close-to-the-event and use of magnetic recorded data.
- **Effort:** Continuous Process Auditing is expected to decrease the amount of procedural effort while focusing on a continuous review of application and audit processes.

## 3. Conceptual View of Methodology

Conceptually, the monitoring system consists of three levels: a data provisioning level, a knowledge level, and a presentation level (Figure 1). The data provisioning level provides the raw data for the analysis. Data can be extracted from operational reports that are produced by the system or through direct data access. The extracted data are stored in a data repository (i.e., a database) and/or stored in raw form.[1] Certain data also need to be stored



CONCEPTUAL VIEW OF METHODOLOGY

knowledge base. This includes informa- bout the structure of the system being ored and analytic definitions. The anal- f the data is performed using various and the output is sent to a presentation

nain elements of the methodology in-

etrics and analytics: metrics are direct easurements of a system. These may in- de things like number of errors, num- r of transactions input, etc. Analytics defined as either functional, logical, or pirical relationships among metrics. ese measurements can be compared ainst system standards and alarms (see low) "fired" if a standard is exceeded.

- standards: level measures determining expected metric and analytic status at a particular point in time.
- alarms: attention directing action, trig- gered by the values of metrics and analyt- ics. Alarms may be hierarchical in nature; i.e., some alarms may just flag an event for inspection, while others may call man- agement attention to serious problems in the system.

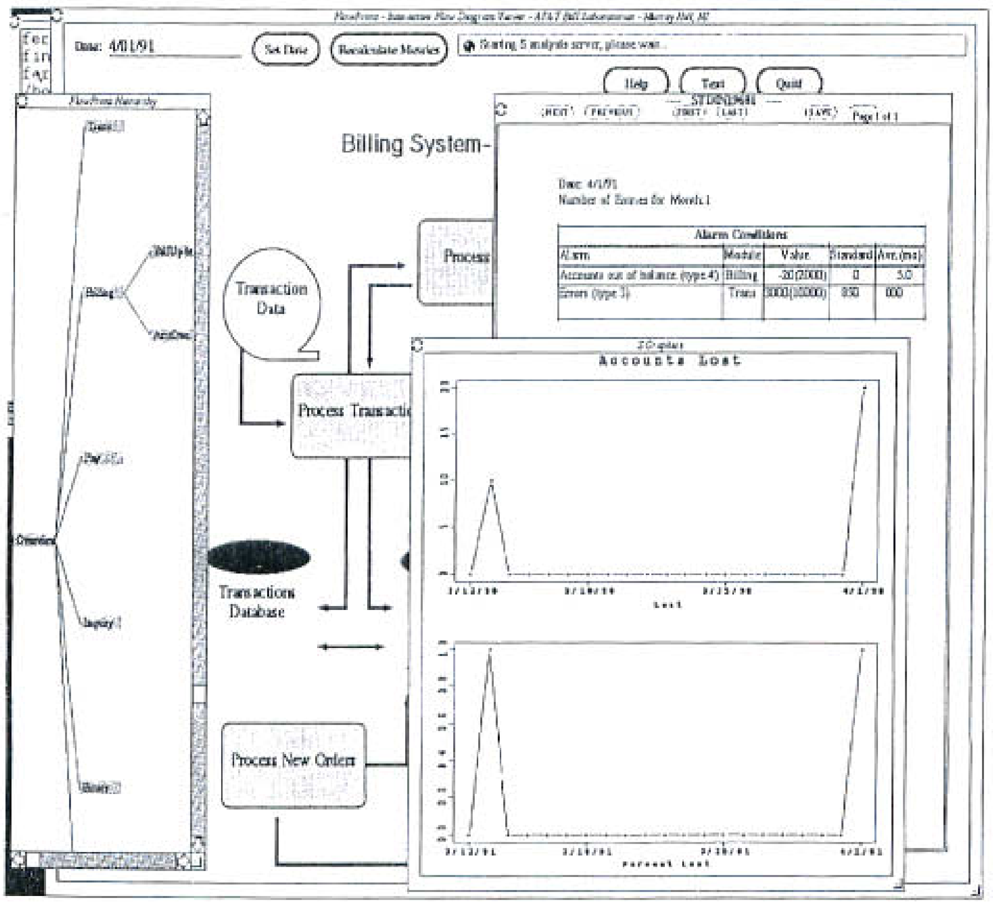
**4. Software Implementation**

Figure 2 was prepared using the CPAS toolkit and has the look-and-feel of any CPAS application. It shows a high-level view of a theoretical billing system. The hierarchy win- dow on the left in the figure indicates what part of the billing system is represented by the flowchart. In this example, the flowchart represents the base node of the billing system hierarchy, i.e., an overview of the system. This node is called "Overview" in the hierarchy window. Other nodes in the hierarchy window correspond to flowcharts that represent a more detailed view of the system being audited. The auditor can use the hierarchy window to move to any flowchart in CPAS by simply selecting the desired node.

Information relating to 4/1/91 is shown in the figure. An alarm report illustrates any outstanding alarm conditions in the system. For example, on 4/1/91, there were two outstanding alarm conditions, an out of balance condition, and an error threshold was exceeded. The report details where the alarm conditions occurred, the actual value of the analytic, an average value for the alarm, and the standard the analytic was compared to. A time series plot shows how many times in the last three weeks the out of balance condition occurred (here, twice). More detailed analytics and metrics relating to the actual billing process and the interface between this module and other modules in the system are found at different levels. This information, taken together, presents an integrated diagnostic view of the system being audited.

"Text," explaining the flowcharts, and "Help," explaining how to use the system, are available at each level. The auditor can print out screens, reports, or graphs at any time for writing his/her audit reports.

Complementing the actual hands-on audit work is an auditor platform, accessible at any level, which can include a series of different



Date: 4/01/91

Starting 5 analysis server, please wait...

Date: 4/1/91
Number of Entries for Month 1

| Alarm Conditions | | | | |
|---|---|---|---|---|
| Alarm | Module | Value | Standard | Ave.(ms) |
| Accounts out of balance (type 4) | Billing | -20(2000) | 0 | 5.0 |
| Errors (type 3) | Trans | 3000(10000) | 850 | 000 |

functions. This platform should ultimately contain at least a statistical package, a graphics package, a spreadsheet package (including a filter to the database), a report generator, and a text editor. These tools can be used for ad hoc analysis or be linked to the "wired-in" procedures in CPAS. An even richer technological environment may incorporate specific **audit document preparation tools** that use high technology hardware to read and interpret printed materials[11] and large amounts of information can be stored and accessed directly using optical disk (WORM) technology.[2]

## 5. Systems Related Issues

The CPAM concept required flexible-modular design and a high degree of flexibility in order to test the concept and prototype the system. Mainframe-based development was deemed too intrusive and too costly. Consequently a workstation-based approach with UNIX-type transitivity and pixel-oriented graphics was chosen.

The CPAS software was implemented under the NeWS windowing system and on a SUN workstation. The NeWS system, at that point (1987), possessed the best set of "widgets" and development tools. It used PostScript as its imaging language and could use a screen, a file, as well as a laser printer as an output medium.

The entire software was constructed using standard UNIX tools with a minimum of low-level programming. Application data were generated in IBM mainframes in the form of standard user reports. These standard system reports were analyzed by a "knowledge engineer" and specific fields chosen for collection. Job Control Language (JCL) specs were included in the application control procedures to specify that a particular copy of a report needed to be sent to a particular distribution node. This JCL specification was the only (and minimal) intrusion in the application.

Once the report was sent to the receiving destination it was placed in an electronic storage bin. A connected UNIX gateway would run periodic (say every 10 minutes) "DAEMONS or CRONS"[3] and capture (snurf) these reports, transform them into mail messages and *mail* (a standard UNIX function) them to the CPAS workstation. Under certain conditions *uucp* (UNIX to UNIX Communication Protocol) was used to transfer the report file to the CPAS workstation. These reports, upon arrival at CPAS, were identified and scanned for the desired data. For example, a report named A121 would be identified as A121.awk,[4] a program scanning routine would be activated, and extracted data placed in a relational database. A commercially available relational database (INGRES[13]) was used as a storage device, separating the data gathering portion of the system from its data analysis and delivery device.

The graphic interface design device was called "Flow-Edit" and is not unlike many graphic design devices now available both in the UNIX[14] and the DOS (e.g. Harvard Graphics) worlds. This tool was used to construct the flowcharts, link the flowcharts hierarchically, and to define different metrics and alarms.

Specific metrics boxes contained data represented to be moving along a flow or contained in a level. These metrics were the result of direct *sql*[15] queries to the relational database. The graphs contained in the windows representing analytics were drawn by a statistical package called *S*.[16] This package was developed for "exploratory data analysis" and contains valuable graphical features. Both the tables and text were generated using UNIX's text editing, formatting and WYSIWIG features enriched by the power of PostScript display on to the screens.[5]

This concept, however, can be extended and implemented piece by piece using standard PC tools. Conceivably, the methodology can be implemented in many different ways, from a pure PC implementation to a full-fledged distributed computing solution with the "audit computer" as the self-contained destination of monitoring/measurement data.

## 6. Discussion

### 6.1 Auditing Issues

The CPAS prototype was tested on two very large financial systems. The first application of the CPAS technology was an evolving system whose features changed rapidly. The idea was to put a prototype in place that contained basic analytics and then work with the auditor, as they used CPAS, to build more expertise into the system. The issue of startup cost to impound the system description into the CPAS platform and the maintenance of the knowledge base became very important. However, the process of knowledge acquisition and recording used under CPAS is not unlike the phases of internal control evaluation and documentation for workpapers that an auditor has to perform. The level of auditor comprehension of the system tends to be deeper under this approach if the auditor (not a system analyst) is to perform knowledge capture.[6]

The CPAS approach probably requires a higher audit startup cost than the traditional audit but the level of audit examination is also consequently deeper and more reliable. The CPAS approach is substantially different from the traditional one and requires balancing of audit evidence and timing of the audit process. Given this, the issue of resistance to change may arise. This can be handled by the issuance of an audit manual that describes how to audit with CPAS and extensive training and technical support of the auditors in the engagement.

### 6.2 System Issues

UNIX provided a useful and flexible platform for the implementation of the concept allowing the utilization of a wide set of generic tools for a rapid development of the prototype. It also worked well in a hybrid environment interfacing with a mainframe-based application. The testing of the prototype indicates its ability to rapidly converge to a full-scale implementation.

The CPAS methodology was developed with special focus on internal auditing but may be extended to external auditing work even for smaller application software (if templates can be developed). CPAS is really a specific instance of the application of a large system monitoring and management technology into an audit domain.

### 6.3 Future Work

Future work will focus on increasing the quality of auditor work by integrating the auditor platform with the auditor workstation, increasing the use of monitoring probes, improving the quality of the auditor heuristics, and impounding more expertise into the system.

Furthermore, future work will expand the decision support features of the prototype as well as some of its intelligence. The main features of such an expansion would include:

- Incorporating AI techniques such as evidence propagation on the belief net.[17]
- Incorporate design analysis methodology based on influence diagrams and probabilistic estimation.[18][19][20]
- Incorporate pattern recognition technology to evaluate time series trends.

* * *

*Miklos A. Vaserhelyi is the KPMG Peat Marwick Professor of Accounting, Graduate School of Management, Rutgers University and Consultant to AT&T Bell Laboratories. Prof. Vasarhelyi's research interests deal with the area of continuous process auditing, textual knowledge structures, and intelligent databases. He has taught accounting and system topics both at the graduate and executive programs in the US, Europe and South America. He has consulted on accounting and computer matters for the government and major firms in the US and Brazil. He received research grants from many institutions. He is the author of 10 books on Accounting Computer Related Topics and over 80 articles in journals and publications.*

*Fern Halper is a Member of Technical Staff AT&T Bell Laboratories in Murray Hill, w Jersey. Her research interests include ntinuous process auditing, dynamic moni- ring systems, and pattern recognition in ge amounts of data. Prior to joining Bell boratories in 1986, she was a member of e Geology Faculty at Colgate University. e has a Ph.D. in Oceanography from Texas &M University.*

*Kazuo J. Ezawa is a Member of Technical aff at AT&T Bell Laboratories. He is an ex- ert in decision analysis and uncertainty in I, especially in the area of influence diagram ased modelling and its related decision sup- ort system development. His current re- earch focus is the use of evidence propaga- ion methodology for advanced pattern atching from database records. He joined ell Laboratories in 1986, and received his h.D. in Engineering-Economic Systems om Stanford University in 1986.*

## eferences

[1]Roussey, R., "The CPA in the Information Age: oday and Tomorrow," *Journal of Accountancy* (October 986), pp. 94-107.

[2]Elliott, R.K., "Auditing in the 1990s: Implications for ducation and Research," *California Management Re- iew* (Summer 1986), pp.89-97.

[3]Vasarhelyi, M.A. and W.T. Lin, *Advanced Auditing* Addison-Wesley Publishing Company, 1988).

[4]Bailey, A. and L.E. Graham, and J.V. Hansen, "Tech- ological Development and EDP" in Abdel-Khalik, A.R. Solomon, I. "Research Opportunities in Auditing: The econd Decade," American Accounting Association: uditing Section, Sarasota, Florida, 1989.

[5]Cash, J.I., A.D. Bailey Jr., and A.B. Whinston, "A Survey of Techniques for Auditing EDP-Based Account- ng Information Systems," *The Accounting Review* (Oc- tober 1977), pp. 813-32.

[6]Wolitzky, J.I., "A Low-Cost UNIX Speech Work- station," *AT&T Bell Laboratories, Murray Hill, NJ. Pri- vate Communication* (July 1985).

[7]Alter, S., *Decision Support Systems: Current Prac- tice and Continuing Challenges* (Addison-Wesley Pub- lishing Company, 1980).

[8]—and K. Hackenbrack, P. De and J. Dillard, "Artifi- cial Intelligence, Cognitive Science and Computational Modeling in Auditing Research: A Research Approach," *Journal of Information Systems* (Spring 1987).

[9]Vasarhelyi, M.A. and F.B. Halper, "The Continuous Audit of Online Systems," *Auditing: A Journal of Prac- tice and Theory* (Spring 1991), pp. 110-125.

[10]Daroca, F.P. and Holder, W.W., "The Use of Analyt- ical Procedures In Review and Audit Engagements," *Auditing: A Journal of Practice and Theory* (Spring 1985), pp. 80-92.

[11]Kahan, S., T. Pavlidis, and H.S. Baird, "On the Rec- ognition of Pritned Characters of any Font Size," *AT&T Bell Laboratories, Private Communication* (January 1986).

[12]Ajo, A.V., Kernighan, B.W., & Weinberger, P.J., "The AWK Programming Language," Addison-Wesley, Reading, Mass., 1988.

[13]Date, C.J., "A Guide to INGRES," Addison Wesley, Reading, Mass., 1987.

[14]Kernighan, B.W. and Pike, R., "The **UNIX** Pro- gramming Environment," Prentice-Hall, Englewood Cliffs, N.J., 1984.

[15]van der Lans, R.F., "Introduction to SQL," Addison Wesley, Reading, Mass., 1988.

[16]Becker, R.A., Chambers, J.M., "S: An Interactive Environment for Data Analysis and Graphics," Wads- worth, Belmont, Calif., 1984.

[17]Shachter, R.D., "Evidence Absorption and Propaga- tion through Evidence Reversals," Uncertainty in Artifi- cial Intelligence 5, P173-190, North-Holland, 1990.

[18]Shachter, R.D., "Evaluating Influence Diagrams," *Operations Research* (1986).

[19]Ezawa, K.J., "Efficient Evaluation of Influence Dia- grams," Ph.D. Thesis, Dept. of Engineering-Economic Systems, Stanford University, Palo Alto, CA, 1986.

[20]Howard, R., "Decision Analysis: Practice and Prom- ise," *Management Science*, June, 1988.

[20]Fukunaga, K., "Statistical Pattern Recognition," Ac- ademic Press, 1990.

[1]Of course, software can be placed directly in the stream of transactions, and monitor the system directly, for any audit concerns.

[2]Software companies, such as Teltrak Advanced Tech- nology Systems Inc., have developed software systems that control access to optional disk storage devices.

[3]These are UNIX-based procedures that self-activate if a particular set of circumstances occurs (DAEMON) or at regular time intervals (CRONS).

[4]Both *sed* and *awk*[12] are pattern scanning languages designed for the identification of specific sequences in text.

[5]The guest-editor's editorial discusses these tools ear- lier in this issue.

[6]In the long range much of this work can be linked to the use of CASE type tools where the knowledge is cap- tured at design and could be easily transported, if not di- rectly used, to the platform.