JOURNAL OF INFORMATION SYSTEMS
Vol. 24, No. 1
Spring 2010
pp. 91–112

American Accounting Association
DOI: 10.2308/jis.2010.24.1.91

Continuous Auditing in ERP System Environments: The Current State and Future Directions

John R. Kuhn, Jr.
University of Louisville

Steve G. Sutton
University of Central Florida

ABSTRACT: Recent research has focused heavily on the practicality and feasibility of alternative architectures for supporting continuous auditing. In this paper, we explore the alternative architectures for continuous auditing that have been proposed in both the research and practice environments. We blend a focus on the practical realities of the current technological options and ERP structures with the emerging theory and research on continuous assurance models. The focus is on identifying the strengths and weaknesses of each architectural form as a basis for forming a research agenda that could allow researchers to contribute to the future evolution of both ERP system designs and auditor implementation strategies. There are substantial implications and insights that should be of interest to both researchers and practitioners interested in exploring continuous audit feasibility, capability, and organizational impact.

Keywords: continuous assurance; continuous auditing; continuous monitoring; enterprise systems; enterprise resource planning (ERP) systems; enterprise risk management; embedded audit modules; management control layer.

I. INTRODUCTION

Over the past 20 years, the discourse on the need for, and ability to deliver, continuous auditing of business information has slowly gained momentum. Today, the concepts of continuous auditing have reached the instantiation stage, becoming a key element in many internal audit departments' risk monitoring strategies. Additionally, continuous auditing is increasingly under consideration as a tool to augment the external audit. This progression has included the evolution of architecturally different methodologies for approaching continuous auditing in computerized environments, primarily embedded audit modules (EAM), which are software modules embedded in an information system (i.e., built into) to monitor activities in such systems (Groomer and Murthy 1989), and a monitoring control layer (MCL), which is an external software module that operates independently of the information system to be monitored but is linked into the system and/or its underlying database to provide a similar level of monitoring (Vasarhelyi et al. 2004). A key component of the discourse in recent years has been focused on the advantages,

Published Online: March 2010

limitations, and feasibility of the two dominant approaches (Alles et al. 2006; Groomer and Murthy 2003; Kogan et al. 1999; Kuhn and Sutton 2006; Vasarhelyi 2006; Vasarhelyi et al. 2004; Woodroof and Searcy 2001).

The constantly evolving state of corporate governance fueled by organizations' focus on strategic enterprise risk management has moved this debate to a focus on practicalities. Continuous auditing tools (EAM and MCL) are quickly becoming key components of overall corporate governance and compliance efforts. Many are designed for specific business processes, enterprise resource planning (ERP), legacy system control settings, transaction processing, and IT processes, etc., that taken together provide the foundation for a more holistic governance, risk management, and compliance (GRC) landscape supporting the entire enterprise (AMR Research, Inc. [AMR] 2008). Continuous auditing applications are being used to support GRC activities across business functions, departments, and IT platforms. As such, third-party software vendors (i.e., non-ERP developers) such as Approva have created additional applications to support enterprise-wide governance interlinking and sub-applications integration designed for monitoring of specific systems and processes. Features of the enterprise-wide governance applications include functionality that supports the identification and management of the varied GRC initiatives across a company (SOX 404 compliance, IT governance, corporate social responsibility, etc.); identification, assessment, categorization, and prioritization of risks; and tracking of key performance and risk indicators in an integrated fashion (e.g., dashboard reporting). The continuous auditing movement has evolved over time, expanded in scope and breadth, and continues to progress forward into unchartered territory.

The ensuing discussion presents a brief history of the origins of continuous auditing and continuous assurance; the development of continuous auditing applications and various alternatives available to companies and auditors; the current state of continuous monitoring and assurance of ERPs in practice; and where we see continuous auditing and overall assurance heading in the future in order to increase the clarity of the current state and provide suggestions for future research that would allow academic researchers to support and lead the growth and evolution. Our perspectives are founded in deep practical experience with ERP systems placed within the context of the evolving theory and research on continuous auditing.[1]

As the push for continuous auditing moves forward, it is important to see where we have been, where we are, and where we are headed. The current discussion contributes to the evolving literature on continuous auditing by examining the *technical* characteristics of continuous auditing through traditional EAM and MCL methods, a modified EAM approach, and newer enterprise-wide/cross-platform applications; the advantages and limitations of alternative methods; and contemporary developments. Of particular interest are the technological advances necessary to support initiatives such as XBRL data tagging and enterprise-wide GRC. Companies of varying sizes and views toward continuous auditing can and will need to choose continuous auditing functionality that supports their individual goals and desires—be it through EAM, MCL, a hybrid approach, or some new adaptation that has yet to evolve.

The remainder of the paper consists of four sections. The first section presents a brief history of the continuous auditing movement; characteristics of various architectural approaches; and advantages/limitations of the approaches from a technical and practical perspective—specifically, real-time monitoring and reporting, complexities of design and maintenance, alarm floods, client

[1] Our views are influenced in part by the first author's personal ERP audit experiences. Prior to entering academia in 2005, the author spent ten years in practice as a Big 4 IT auditor (managed the SAP audit of WorldCom during the fraud restatement) and Financial Systems Manager (managed an SAP implementation), and as an Internal Audit Manager at Siemens Corp.—auditing, evaluating, and implementing continuous auditing functionality both pre- and post-Sarbanes-Oxley 404.

independence, and external auditor legal liability. The second section provides an overview of existing continuous auditing software developed by third-party vendors and ERP developers—primarily, a look at SAP (ERP developer) and Approva (third-party) ventures into enterprise-wide continuous auditing to support overall GRC. The third section discusses continuous monitoring and auditing issues in need of research at both the application and enterprise-wide level that require a variety of research methods. The final section provides a brief summary of this review on the current status and future directions of continuous audit integration with ERP systems.

II. CONTINUOUS AUDITING BACKGROUND AND TYPES OF SYSTEMS

Beginnings of Continuous Auditing and Development of Embedded Audit Modules

ERP systems designed for the processing of business transactions traditionally have been built upon a core database management system (DBMS). Groomer and Murthy (1989) raised the awareness and interest of the accounting research community in the notion of EAM as a viable approach to supplement and enhance the audit of companies with increasingly complex computer-based accounting systems relying on DBMS. They described an approach to continuous auditing of database-driven accounting applications using EAMs to address the unique control and security aspects of database environments. They defined EAM as "modules (code) built into application programs that are designed to capture audit-related information on an ongoing basis." Braun and Davis (2003) similarly define EAM as follows:

> This technique involves the auditor inserting an audit module in the client's application that will identify transactions that meet some pre-specified criteria as they are being processed … Often, these modules are designed in such a way that they can be turned on and off, reducing costs but also reducing coverage. (Braun and Davis 2003, 726)

Vasarhelyi and Halper (1991) advanced the work of Groomer and Murthy (1989), developing the Continuous Process Audit Methodology (CPAM) and illustrating the design and functions of an EAM in a hypothetical customer billing system. The authors subsequently tested a prototype continuous auditing system they created based on CPAM for use with three large financial systems. The prototype results indicated a deeper and more reliable level of audit examination was achieved.

The early works of Groomer and Murthy (1989) and Vasarhelyi and Halper (1991), in conjunction with the joint report on continuous auditing sponsored by the Canadian Institute of Chartered Accountants and American Institute of Certified Public Accountants (CICA/AICPA 1999), spawned a stream of continuous auditing research. From an architectural standpoint, a good portion of the efforts adhered to the methodology from these two early works by focusing on EAM as the underlying technological approach (Groomer and Murthy 2003; Debreceny et al. 2003; Murthy 2004; Debreceny et al. 2005; Chen et al. 2007; Loh and Jamieson 2008).

Debreceny et al. (2005) defines a set of essential characteristics for successful continuous monitoring applications: (1) an end-user environment that allows the auditor to establish a set of queries to test transaction integrity constraints either from a pre-defined suite of queries, the modification of the attributes of pre-defined queries, or by the creation of new queries by the construction of simple scripts; (2) a process for registration and scheduling of these queries; (3) a method for running these queries against the flow of transactions for violations either continuously or temporally; (4) a capacity for reporting violations electronically; and (5) an ability to copy the transaction details of violations to secondary storage. We elaborate on these as we create a comparison of characteristics for a variety of continuous auditing application design approaches. But first, clarification of some related terms will help. *Continuous monitoring*, also often referred to as *continuous auditing*, consists of the analysis of data on a real- or near real-time basis against a set of predetermined rule sets. Henrickson (2009) suggests the delineating determinant between the

two can be viewed as monitoring being a management responsibility and auditing being an auditor's responsibility. *Continuous reporting*, on the other hand, is the continuous reporting of information about defined phenomena and in a continuous auditing context includes the notification of rule violations occurring as a result of continuous monitoring. Finally, at the macro level sits *continuous assurance*, as noted by Alles et al. (2002, 128):

> [Continuous auditing] is best described as the application of modern information technologies to the standard audit products … Continuous auditing is another step in the path of the evolution of the financial audit from manual to systems-based methods … By contrast, continuous assurance sees continuous auditing as only a subset of a much wider range of new, nonstatutory products and services that will be made possible by these technologies.

All of the approaches (monitoring, auditing, assuring) require the same basic technical capabilities. There are alternative approaches to delivering these capabilities, however, other than EAM. Before considering these alternatives, we review the design and implementation characteristics of EAM.

First, with EAM the programming code for the audit procedures/tasks is developed and implemented *inside* the walls of the target application (i.e., embedded) using the programming language of the application itself. For instance, SAP developed a unique programming language called ABAP for specific use with their core ERP software. If, for instance, a company wishes to build an EAM audit check that verifies every invoice has a corresponding purchase order and goods receipt with identical quantity and dollar amounts within a certain threshold range (i.e., commonly referred to as the three-way matching process), programmers for the company can create an ABAP program and literally "plant it" inside SAP alongside the programs delivered with SAP. This new program is considered "non-native code" in the respect that it is added to SAP afterward. Second, EAM audit functionality can evaluate transactions against programmed audit criteria *live* (i.e., real-time monitoring) as they happen in the application. That does not necessarily mean the EAM module *has* to run constantly, but it *can*, if so desired. Third, EAMs include reporting functionality that notifies predetermined individuals through any of a number of options, including emails, pager notifications, system reports, etc., of any transactions violating the audit procedures. Due to the real-time monitoring functionality, EAM consequently offers real-time reporting. In our three-way matching example, if an invoice was processed and flagged for payment but the invoice amount exceeded the original purchase order by a predetermined threshold (say 1 percent), then an instantaneous notification (referred to as an *alarm*) would be sent to individuals in the internal audit department or external audit firm (or both) depending on what party relies on the EAM. Fourth, storage of the alarms for data retention purposes resides in the same databases/database structures with normal transactions so regularly scheduled application backups capture the EAM auditing data. This also allows the auditor to handle real-time alerts in a batch mode, if so desired. We explore the rationale for such a decision later.

Alternative Technical Architectures to Traditional EAM

The traditional EAM model prescribes coding of audit procedures directly into the auditee's host system to facilitate real-time monitoring and reporting of transactions and system settings. However, certain characteristics of a traditional EAM instantiation may not be the best fit for every company and situation. Therefore, the question arises as to the alternatives available to companies and auditors for continuous auditing implementation. Can the EAM model developed in extant research be modified? Are other architectures more feasible under certain circumstances? Or, are the various continuous auditing architectural strategies situational and, therefore, amenable to co-existence within organizations?

EAM Ghosting

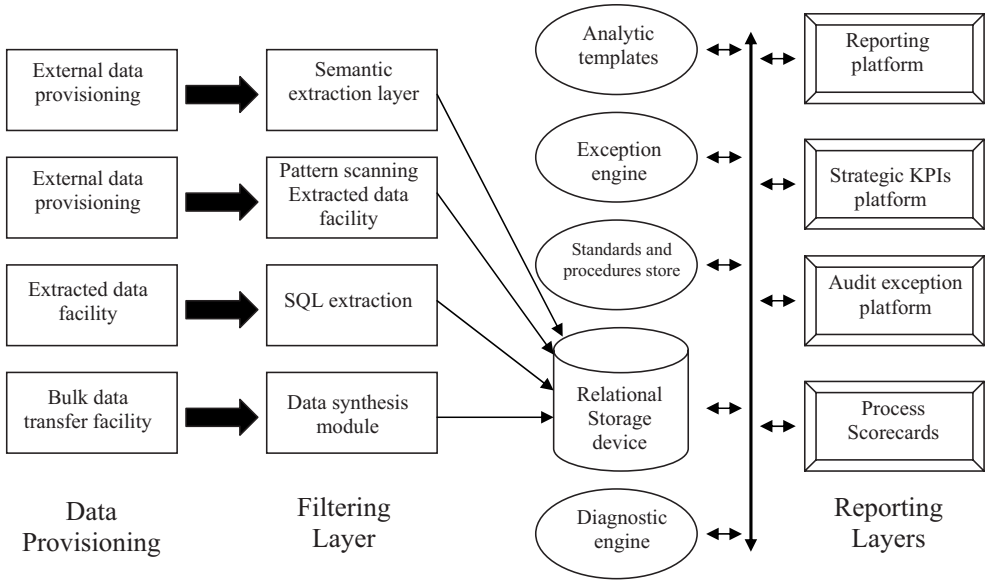
System "ghosting" offers an opportunity to benefit from the advantages of EAM yet implement, operate, and maintain audit functionality outside the production system of a company keeping the audit program separate from the live system. Ghosting entails operating a "copy" of an entire system on separate hardware, including data and system settings, in a real-time fashion similar to an instantaneous fail-over common to disaster recovery and business continuity planning procedures and redundant firewall systems. The EAM functionality would reside in the ghosted copy where there would be no risk of affecting live transaction processing occurring within the production environment.

Two slight variations of EAM Ghosting also exist. Many companies operating large-scale ERP systems utilize a stream of copies of the production environment used for development and change management. Separate systems exist for development (DEV), quality assurance (QAS), and production (PRD), generally hosted on different physical servers that communicate through transport management software (VMware 2009). Changes intended for production will first be configured and tested in DEV by IT personnel, then migrated to QAS for end-user testing before transport to PRD. Typically, the DEV instance contains production data a few days to months old, but with current systems settings. QAS data will be closer in time to PRD with the same system settings. Both variations take advantage of the fact that the DEV and QAS physical servers experience significantly less usage than PRD. For the first variation, companies can segregate a physical server such as the one housing QAS into two partitions, one supporting QAS and the other supporting a mirror-image of PRD updated nightly (rather than real-time) that contains the EAM code. The primary advantage of this approach over standard EAM Ghosting is that IT personnel do not have to worry that real-time processing is updated instantaneously in the system housing the EAM. Second, virtualization technology provided by companies like VMware can take advantage of unused physical hardware space and create an independent virtual machine on the unused QAS server that ghosts the production system but contains the EAM programs. Virtualization requires less physical hardware space and less memory to operate than creating a separate partition on an existing server.

Monitoring Control Layer

Vasarhelyi et al. (2004) introduce an alternative continuous auditing system architecture to EAM referred to as the Monitoring and Control Layer (MCL). MCL can be viewed as the next stage in the evolution of continuous auditing application design—not necessarily one subsuming EAM but more as an alternative to cater to different circumstances. The MCL approach takes the continuous auditing system and "hooks" it into the client's existing enterprise system, generally using a middleware layer to integrate loosely coupled applications such as ERP systems, legacy systems, and web-based applications (e.g., supply chain management, customer relationship management, etc.). The main elements of the MCL architecture consist of: (1) data capture layer; (2) data filtering layer; (3) relational storage; (4) measurement standards layer; (5) inference engine; (6) analytic layer; (7) alarms and alerting layer; and (8) reporting platform as depicted in Figure 1 (Vasarhelyi et al. 2004). Essentially, the continuous auditing system (designed with a simple user interface and underlying database reusable for multiple clients) resides outside the client's network environment (for external auditors) and is controlled by the auditor. The continuous auditing system receives periodic interfaces of client data as determined necessary by the auditor (i.e., not real-time) that are processed against a predefined rule-set of audit procedures inside the continuous auditing application, which is physically and virtually outside the client's audited system (unlike EAM). Any violations as defined by the rule-set trigger an automatic alert to the auditor. These alerts are stored inside the continuous auditing application under direct control by the auditor, not inside the databases supporting the client's ERP system.

FIGURE 1
The Architecture of the MCL Layer
Continuous Monitoring and Control Platform



Source: Vasarhelyi et al. (2004).

Recent research efforts by Kuhn and Sutton (2006), Alles et al. (2006), and Alles et al. (2008) offer evidence of the viability of an MCL continuous auditing approach in an ERP environment. As an example of how fraudulent activities at WorldCom could have been detected earlier, Kuhn and Sutton (2006) lay out the design specifications for integrating a specific set of metrics appropriate for the WorldCom financial reporting system that could have been used to continuously monitor transactions. The study demonstrates how financial transaction data (i.e., journal entries) can be extracted from the client database layer without any direct processing inside the client system. Both Alles et al. (2006) and Alles et al. (2008) document a functioning system prototype developed at Siemens for the continuous monitoring of business process controls and the detection of exceptions to those controls. The prototype demonstrates the use of ABAP (SAP's unique language programming) to extract the business process control data from Siemens' ERP system. The two studies differ on several key aspects. Kuhn and Sutton (2006) *design* continuous auditing procedures for the testing of *financial transactions* based on an *historical case* of financial reporting fraud. Alles et al. (2006) and Alles et al. (2008), on the other hand, *implement* a continuous auditing system for the testing of *internal controls* in a *live* environment.

Table 1 presents general characteristics for continuous auditing systems and how various architectural types compare on those traits. The remainder of this section focuses on analysis of the issues and concerns associated with each approach, including some that all approaches share.

TABLE 1
Characteristics of Continuous Auditing System Types

Characteristic	EAM	EAM Ghosting	MCL
Contains predefined audit rule-set.	Yes	Yes	Yes
Location of audit rule-set and program.	Inside target system	Outside target system, inside client network	Outside target system; outside client network if external auditor; inside client network if internal auditor
Requires installation of additional hardware to implement.	No	Possibly	Yes
Performs real-time monitoring.	Yes	Yes	No
Performs real-time reporting.	Yes	Yes	No
Automatically notifies auditor of rule violations.	Yes	Yes	Yes
Location of stored rule violations.	Target system database	Database external to target system, inside client network	Database external to target system; outside client network if external auditor; inside client network if internal auditor
Owner of system resources.	Client	Client	Client if internal auditor, Audit firm if external auditor

Limitations of Continuous Auditing Applications

Debreceny et al. (2005) provide an overview of several limitations in the use of EAM along with observations regarding the lack of perceived demand for EAM capability. In the Alles et al. (2002) discussion on the feasibility and economics of continuous assurance, the authors raise some concerns with EAM for external auditors. They question whether EAMs designed by external auditors and incorporated into ERP systems ultimately transform the ERP into an "unauditable" system due to a perceived or actual lack of independence. A technical dependence invariably exists between the assuring system (EAM) and assured system (ERP), and the auditor may well be perceived partially responsible for the client's enterprise system. As Alles et al. note, questions such as these can only be answered through theoretical and empirical research. The following discussion expands upon the limitations of EAM, EAM Ghosting, and MCL in greater depth in an effort to improve the understanding of the evolution of continuous auditing and related applications, specifically why ERP vendors were slow to develop EAM functionality.

Technical Concerns

ERP systems, by the nature of their integrated processes, require significant information system resources to operate at an optimal level. As Debreceny et al. (2005) noted, Vasarhelyi and Halper (1991) expressed concerns related to the computing resources EAMs require to monitor transactions and system settings in real-time. Those additional resource requirements risk slowing

the overall processing of the ERP system dramatically. Debreceny et al. (2005) note that performance concerns might be addressed by adding appropriate hardware and software resources, although associated costs would occur. Those costs could be significant and ongoing (e.g., additional leased space in an outsourced data center).

Beyond the substantial monetary cost of additional hardware and software to support the increased processing requirements, a company also faces the potential that purchased hardware and software may not resolve the problem. Rather than being primarily a problem driven by hardware capabilities and power, the fatigue on ERP applications is derived heavily from the impact of *non-native* code. EAM is rarely included as a native component of an ERP system in the early development stages and therefore generally requires either subsequent programming to build in the audit functionality or integration of an add-on module. Even when using the ERP's language (e.g., ABAP for SAP R/3) relatively small program modules running in the background can have rather severe adverse effects on the transactional processing efficiency of systems along with other related performance issues. The first author personally experienced this when turning on the logging functionality in SAP per the recommendation of the company's auditors, and this was *native* code. The first author, at that time the financial systems manager responsible for the configuration of the SAP system at Siemens Power Generation Corp., promptly turned off logging and the overall system performance immediately increased. In addition, when subsequently serving as an external IT audit manager, the first author noted similar situations with other companies' SAP instances. In a non-random sample based on the author's client portfolios, out of 12 SAP implementations audited over a two-year period, none of the organizations chose to use the embedded audit technology for monitoring configuration changes. The client's standard response to associated management review points was that the embedded functionality would bring the software to a grinding halt. Henrickson (2009) noted that this situation persists with SAP today and companies broadly refuse to use the embedded application.

EAM Ghosting retains the integrity of the production system and, with virtualization software, reduces technology costs of implementing ghosting. However, the concern remains as to whether *non-native* code adversely affect transaction processing in the ghosted system to a point where the system trudges along and possibly crashes itself? A logical response is to push for the module to be included in core functionality of the production or ghosted system and stress tested, but even in this case it will not be similar to the other *switches* that can be activated in ERP software for configurability to business processes as these *switches* generally relate to database views. EAM modules require programmed procedures that must be specifiable by the auditor based on testing interests at a given point in time and require runtime processing capability to operate on an ongoing basis in the background. For EAM ghosting to be a viable alternative, significant testing of EAM functionality in a variety of hardware and data conditions would need to occur.

Practical Issues

Design and maintenance concerns also arise when implementing EAM into an ERP environment, specifically in large organizations operating multiple instances of ERP applications. EAM functionality, by nature, must be integrated directly into the underlying programming code. Therefore, a company utilizing more than one ERP would be forced to design, test, implement, and maintain the EAM coding in each individual application thus consuming valuable resources. For example, the continuous monitoring pilot project initiated by the Siemens Internal IT Audit Department (Alles et al. 2006; Alles et al. 2008) encompasses the audits of the SAP instances (the standard ERP companywide) for all the operating companies located in the United States. For Siemens, this would mean dealing with over 20 different SAP instances; one division alone, Siemens Power Generation, operates three SAP instances with a development, testing, and pro-

duction environment for each in addition to several sandboxes—all of which would require individual maintenance of EAM code. For several reasons, including the ability to manage audits of disparate systems located and maintained by distinct and separate IT organizations, the Siemens project team chose to implement an MCL solution that could be controlled outside of the SAP instances with feeds from those instances directly into the MCL application.

An additional concern often raised in the research literature (i.e., Kogan et al. 1999; Alles et al. 2006; Kuhn and Sutton 2006; Alles et al. 2008) relates to information overload from alerts when implementing continuous auditing systems that could mask existence of underlying problems, regardless of the architectural strategy employed (EAM or MCL). The initial volume of "false" alerts, commonly referred to as "alarm flood" could be overwhelming until the logic of the alert monitors has been modified during the period immediately after the system goes live (Alles et al. 2006; Alles et al. 2008). The maintenance issues noted become an issue here in an EAM implementation given that the ERP system would need to have the EAM modifications put in place after the ERP system is in a "go live" mode of operation. Such an alteration during the operation of the ERP to support business operations is likely to give more than one company manager trepidation as to the value of the continuous auditing in terms of the risk/benefit trade-off.

Even after adjusting the alert systems for any type of continuous auditing application, the audit process still poses an information overload dilemma for the majority of companies. The month-end and year-end closing processes of the accounting information systems to prepare the financial statements for reporting typically involve several days of analysis, posting of adjusting journal entries, and verifying the successful data transmission from legacy interfaces. The books are not complete until this process occurs. False alerts and notifications to auditors would be generated throughout the month on certain types of incomplete data raising misguided questions and resulting in unproductive time spent investigating non-issues. To be efficient (and manageable), a continuous auditing system needs to allow the auditor to dynamically adjust metrics, turn off the monitoring during periods where certain accounts may be in flux and the auditor is not interested in the adjusting and correcting entries, or as certain accounts fluctuate based on the normal business cycles of the client. For example, while current assets and liabilities will typically be monitored on a continuous basis, noncurrent and long-term assets and liabilities may be monitored in a much different way—including a shortened periodic snapshot versus true continuous auditing. This would be particularly concerning to the external auditor using EAM because the auditor would need to coordinate with client IT personnel to make any changes, whereas auditor direct control over the MCL application allows quicker response time and requires less time from the client (an issue with any audit).

Additional External Auditor Concerns

The inherent design of the EAM architecture contains a plethora of critical deterrents to adoption by external audit firms. The most critical are likely to be issues related to system design and maintenance, client independence, and legal liability. Here, an MCL approach provides some distinct advantages over EAM and Ghosting for external audit purposes.

System design and maintenance. As proposed by Debreceny et al. (2005), the external auditor would design and maintain the EAM functionality in the client's accounting information system to ensure independence and prevent client knowledge of the audit approach. A number of practical concerns arise for the external auditor when considering EAM and EAM Ghosting.

First, in the strictest interpretation of the external auditor's role, the auditor would require access to the client's systems and the client would need to be willing for the auditor to make unsupervised changes. Given the experience most companies have had with poorly designed systems, lost data, failed conversions, and delayed identification of processing bugs, most clients are likely to be terrified of the idea of surrendering control to the auditor—particularly when the

benefit to the client of continuous external monitoring may not be clear. The alternative is for the auditor to make changes, but to supplement the efforts with internal corporate staff that could facilitate implementation. This would require the auditor to rely on the availability of client personnel to assist in the design, testing, and implementation of code changes at the same time that the auditor is attempting to maintain the integrity of the audit procedures. Obtaining client personnel time historically has been a challenge all firms face with the traditional assurance model. Increasing the requested time away from a client's normal business activities, particularly the client's high-paid IT specialists, would likely be a major challenge.

Second, as noted by Debreceny et al. (2005), extensive knowledge of ERP programming languages by the auditor is required. Debreceny et al. (2005) effectively highlight the complexity of ERP systems, providing an example of the tables and attributes included in a single SAP instance. Using an EAM approach, an audit firm will need to hire and train auditors to not only be proficient auditors, but also knowledgeable in the diverse languages of the various ERP systems used by clients. Acquiring quality IT auditors, particularly in the post-SOX era, has been a major challenge in and of itself (Ernst & Young 2006). Internal audit departments, the PCAOB, and academia all compete against the auditing firms for the small pool of qualified individuals in the current environment. Less than 50,000 Certified Information Systems Auditor licenses have been awarded globally. Adding significant programming skills to the position requirements creates a potentially insurmountable challenge in the recruiting and staffing process.

Finally, coding for audit procedures using EAM functionality resides in a client's host system. Employing EAM continuous auditing throughout an audit firm's client portfolio requires significant resources to design and maintain audit procedures for each client that offer limited or no scalability/reusability for other client engagements. Such modules may have replicability in other client instances of the same ERP software, but in many cases clients have multiple ERP systems operating simultaneously that will each require the design and implementation of an EAM module—potentially all in different programming languages.

The MCL approach worked well for the Siemens internal department but, on the flip side, how would external audit firms operationalize MCL? As explicitly stated by a Deloitte and Touche LLP partner at the 2005 World Continuous Auditing and Reporting Symposia (WCARS) held at Rutgers University, public accounting firms are hesitant to embrace implementing continuous auditing solutions for the audits of their ERP clients (Fogarty 2005). For MCL applications the firms would need to acquire significant IT resources (hardware, software, and IT personnel) to develop, maintain, store, and secure continuous auditing applications in-house for each client. Are the applications scalable across clients and/or industries? Does the cost justify the effort and costs? EAM and MCL pose many of the same design and maintenance challenges for external auditors.

Client independence. Debreceny et al. (2005) recognize several of the security challenges, from both the client and auditor viewpoints, that arise from designing and maintaining the underlying logic of EAM in the client's system. External auditors cannot reasonably expect system administrators to allow the auditor the ability to maintain functionality in the system with the administrator having no permissions to access the code in the event of system performance or processing issues.

Ironically, the same change management procedures IS organizations must adhere to in order to maintain compliance with prescribed control procedures under SOX would require proper testing and authorization of the auditor's changes to the systems since the changes impact the processes related to financial information and financial reporting. Should the auditors abide by a different set of rules and be exempt from such quality assurance testing? If not, who would be responsible for enforcing the change management procedures and documentation?

Allowing client personnel access to the EAM code during the design and implementation stage and *ex post* to implementation addresses these two questions but poses an additional prob-

lem, that of independence. Both scenarios would result in knowledge of the audit procedures and ultimately could be subject to client manipulation. The client could potentially utilize the inside information on the auditor's monitoring procedures to conduct fraudulent activities in unaudited areas or design suspect transactions configured intentionally to not trigger alerts. In short, the existing independence requirements and prescribed procedures for unannounced audit visits are in place to assure the auditor has the opportunity to monitor and search for fraudulent activity without the client having *a priori* knowledge of what will be tested and observed. Sacrifice of this ability seems atheoretical to the underpinnings of audit independence.

Legal liability. A major deterrent to auditors assuming many of the responsibilities often seen desirable by the public (e.g., fraud) has been the risk of legal liability. While the client may be apprehensive about an auditor having access and making changes within their system or simply guiding the implementation of EAM modules within the system, the auditor would likewise be expected to have some trepidation over the risk involved in making such changes to a client's system. Systems are core to an entity's ability to continue its operations and compete in the marketplace. The threat to systems from interacting with trading partners has become severe enough that some powerful corporations such as Wal-Mart have required trading partners that interact with their systems through e-commerce to sign a contractual agreement that the trading partner is responsible for all costs related to damages and repairs to Wal-Mart's systems along with all costs incurred from lost business should their system be infected or otherwise damaged by the trading partner linkage (Gerhardt 2002). It does not seem unreasonable to assume a client organization would want a similar level of risk reduction from the auditor if the auditor is going to be put into a position where damage to the client's system may occur.

Regardless of whether the client and auditor have signed a damage responsibility agreement, should any of the modifications implemented by the auditor adversely affect the processing of client business transactions, the client would likely take legal action against the auditor in order to recover damages incurred. This threat of litigation alone would likely be sufficient to cause many public accounting firms to refuse to undertake continuous auditing of client systems using an EAM strategy.

The GRC Software Market

The massive amounts of resources (man-hours and money) expended to comply with SOX during the first years have driven companies to seek opportunities to streamline the ongoing compliance process. Continuous auditing applications offer the ability to strengthen the internal control environment and provide efficiencies in the overall compliance activities. However, as Debreceny et al. (2005, 23) noted, ERP software vendors generally "believe customers are unwilling to pay a premium for a function that is not perceived as mission-critical" and, therefore, only limited (if any) built-in functionality exists. The disconnect between ERP vendors and their customers may lie in the fact that accelerated filers initially grossly underestimated the resources required to comply in 2004 and the ERP vendors did not receive any type of substantial call for the additional functionality early in the SOX compliance process.

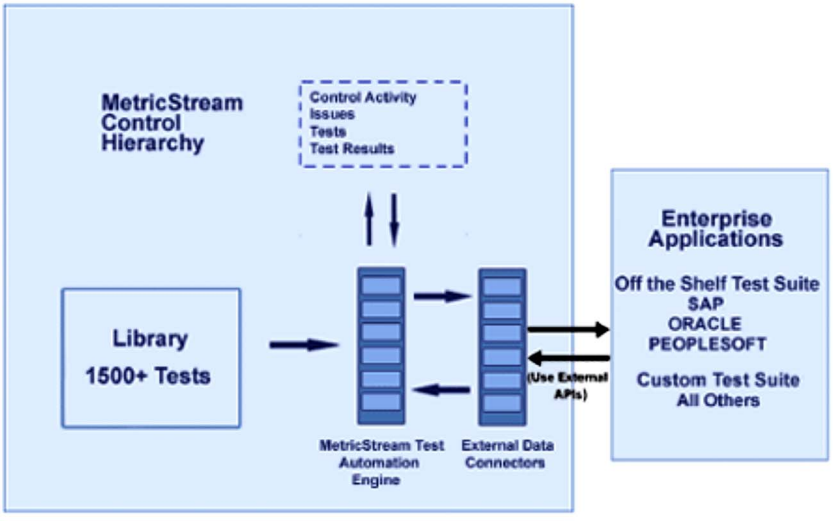
Companies are now looking for ways to reduce the level of manual effort and the related consulting costs incurred the early years of SOX 404 compliance through technology applications. Furthermore, recent PCAOB Auditing Standard No. 5 guidance that recommends external auditors rely more on internal audit work provides an even greater impetus for companies to find alternative solutions for performing automated testing. Independent, third-party software developers have seized the opportunity and developed SOX compliance tools that monitor the effectiveness of internal controls in the financial systems along with the system settings that facilitate control processes; the vendors target these applications directly at internal audit departments and other

corporate groups. These third-party applications are based on the MCL continuous auditing architecture as they operate externally to and interface with the targeted system to obtain the data to be audited inside the MCL application; the audit results are then stored outside the audited system.

Evidence of a heavy emphasis on MCL development is prevalent among the numerous software vendors presenting their products at the 2005 WCARS. For instance, representatives from five vendors at the conference, including industry mainstays such as ACL and CaseWare IDEA each demonstrated the capabilities and technological aspects of their respective software, including the MCL-based capability for supporting continuous auditing of internal controls within ERP environments (Rutgers Business School 2005). Other leading vendors with continuous auditing compliance tools include Approva, MetricStream, Oversight, and Trintech (AMR 2008). Figure 2 presents the MetricStream architecture for their Test Automation Engine (an MCL application) that connects to various ERP applications (i.e., the targeted system) via application programming interfaces (API). The AMR Research Enterprise Performance Report on the GRC landscape thoroughly analyzes 24 third-party vendor applications designed for GRC including continuous auditing functionality. Table 2 lists a subset representing the major vendors operating in the continuous audit space, as noted in the AMR report, along with an overview of available functionality.

The array of vendors, including several well-established software companies, provides evidence that there is demand beyond that served by ERP vendors for the establishment of software-based continuous auditing capability in enterprise system environments. Companies appear to be quite interested in identifying software solutions for continuous auditing that will help reduce the

FIGURE 2
MetricStream Architecture



Source: MetricStream website on July 7, 2009. Available at: http://www.metricstream.com/solution_briefs/Automated_Testing_Internal_%20Controls.htm.

TABLE 2
Third-Party Continuous Auditing Solutions

Functionality	ACL	Approva	CaseWare IDEA	MetricStream	Oversight Technologies	Trintech
Predefined Analytics	Yes	Yes	Add-on	Yes	Yes	Yes
Automatic Notification and Workflow Management	Yes	Yes	Yes	Yes	Yes	Yes
ERP Compatibility	Yes	Yes	Yes	Yes	Yes	Yes
Cross Application Integration	Yes	Yes	Yes	Yes	Yes	Yes
Audit Financial Transactions	Yes	Yes	Yes	Yes	Yes	Yes
Business Process Controls	Yes	Yes	Yes	Yes	Yes	Yes
Access Controls	Yes	Yes	Yes	Yes	Yes	Yes
Audit System Configuration Settings	Yes	Yes	—	—	—	—
Risk and Control Framework	—	Yes	—	Yes	—	Yes
Risk Management	—	Yes	—	Yes	—	Yes
Dashboard and Reporting	—	Yes	—	Yes	—	Yes

costly effort of SOX compliance activities through technology solutions. The early successful ventures appear to have a clear preference for MCL-style continuous assurance, in some part due to the many issues raised previously.

Further evidence of the changing dynamics of the marketplace is SAP's acquisition of Virsa Systems (Martens 2006) and SAP's pitch (along with its implementation partners) to bundle Virsa's SAP Compliance Calibrator with its SAP ERP applications as a SOX compliance tool (SAP 2007; Deloitte 2007). Virsa's Compliance Calibrator (VCC) facilitates a range of control activity monitoring with a particular focus on analyzing the specification of segregation of duties within an SAP implementation. SAP touts VCC as "the only product that effects real-time, 24/7 compliance by stopping authorization violations before they occur" and "allows you to test your entire IT landscape for SoD violations and monitor critical transactions in real-time" (SAP 2008). Not only does VCC clean the IT landscape via real-time scanning, but keeps it clean through real-time simulation by conducting "what-if" analyses on new security access roles (e.g., accounts payable, purchasing, etc.) in the development environment to identify potential violations before they migrate to the production environment. VCC supports the cross-enterprise risk analysis of any IT application (SAP, Oracle, legacy, etc.) from a single dashboard. This latter feature of VCC represents the push in practice for organizations to not only have continuous auditing of key applications, but enterprise risk management solutions that provide company management a clearer picture of overall corporate governance and GRC. As can be seen in Table 2, third-party vendors such as MetricStream see broader governance management as the way of the future. In a recent conversation between one of the authors and a Big 4 IT audit partner, the partner commented that IT audit hours are dropping significantly as external auditors rely more on internal audit and client-operated automated controls and governance testing mechanisms. Audit manager hours are increasing slightly but staff hours are declining rapidly as less and less controls testing is performed by external IT auditors. The transition to enterprise-wide governance technology

tools and increasing reliance on company controls and testing introduces interesting research questions for the future state of continuous auditing and the potential impact on external audit quality.

III. RESEARCHING TO SHAPE THE FUTURE

While our prior discussion highlights the current energy within both the academic and practice communities as solutions to problems surrounding efficient and effective continuous auditing processes are sought, the reality is that there are few answers. There are prototype systems, promising instantiations, and prescriptive frameworks. There is also an array of frequently raised concerns. While these works yield insights, they mostly raise more questions than they provide answers. As Behn et al. (2006) note, the move to a continuous auditing model will be more *evolutionary* than *revolutionary*. In this section, we explore the array of unanswered questions.

Opportunities for Research on EAM

Debreceny et al. (2005) note the major problems with EAM are technical and that these issues may diminish as hardware and software evolve and improve. However, care should be taken in focusing on the problems of today in a changing technical world where advances may not necessarily resolve the current problems of EAM. For instance, SAP is currently moving its underlying ERP data architecture away from relational database technology to web-based technologies leveraging XML data tagging (Taylor 2008). Rumors were rampant and later confirmed by executives that the delayed release of SAP's new system has been related to system response times increasing instead of decreasing, as was originally expected with the new architecture (Greenbaum 2008). On the surface, this does not bode well for EAM supporters. From an academic researcher perspective, however, it suggests there is a great need to reconsider how EAM systems operate in these new environments. This provides our first research challenge. Most likely, design science work is best suited to take on this challenge.

Challenge 1: How can EAM be efficiently implemented within an environment using XML data formats to store and retrieve enterprise information?

Timeliness of this research may also help avoid the pitfalls EAM experienced in first generation ERP systems. As noted earlier, these first generation systems provided EAM capability as an add-on to the system after the core architecture existed. This is considered a major cause of the system inefficiencies that have relegated EAM solutions as non-usable in current ERP systems. Integration of best practices for EAM monitoring as a part of base system configuration could improve efficiency by integrating EAM into core ERP system functionality. Research is needed, however, to better understand potential best practices and how they can be integrated into the core of XML-based systems.

From an external audit perspective, there are additional research issues beyond the technical. First, it has been assumed that EAM systems would create independence problems for external auditors. However, while there has been extensive prior debate in the academic literature over independence issues surrounding external audit and the various consulting services that audit firms offered in a different era, this debate has not teased out the issues surrounding EAM use by external auditors. This presents two challenges—the need for a descriptive stream that explores the issues under current regulations and a normative stream that explores what the possibilities could be and how the public interest would best be served.

Challenge 2: What are the independence issues and interpretations that relate to EAM implementation? Are there forms of EAM that would meet independence requirements?

Challenge 3: Considering independence as a concept designed to protect the public interests, what are the conceptual independence constraints that must be considered in applying continuous audit technologies? How does continuous auditing fit into the auditor's role of serving the public interest?

Similarly, the legal liability issues have largely been relegated to a discussion of the potential impact from damage to the client's system. There are also other legal liability issues that could affect the external auditor. Past research looking at the auditor's use and non-use of available information technology suggests that the auditor may be given more leeway in the courts when implementing new technologies than when there is a failure that might have been detected if a new technology had been implemented (Sutton et al. 1995). The technologies upon which this past research was based are much different from what we see with continuous auditing technologies. Research that re-examines this issue within the spectrum of continuous auditing and evolving social norms as related to technology could provide both a better understanding of the legal issues and provide guidance to practice. There is also a stream of research that considers how case particulars impact juror decision-making (e.g., Reckers et al. 2002). Litigation is certainly a major concern for external auditors, and a better understanding of how jurors might react to continuous audit technologies and approaches would be welcome and may open opportunities for continuous audit use by external audit firms.

Challenge 4: What will auditors' liability be for the use of continuous auditing when they fail to detect material misstatements or frauds? If the continuous auditing technology is the source of the problem, will liability be less based on the auditor attempting to use state-of-the-art technologies? Is this liability greater when an EAM approach is used?

Challenge 5: How will jurors react to auditors' use (non-use) of continuous audit technology when there is a potential audit failure? Is the auditor's liability increased when using EAM versus other continuous audit techniques?

Although the majority of continuous audit research and much of our own discussion on the limitations of EAM has focused on the ramifications to the external audit, opportunities exist to extend research to the internal auditing domain. PCAOB AS No. 5 places an increasingly greater reliance on the work of internal audit departments as evidence usable by external audit firms in order to reduce duplication of effort and subsequently audit costs. Many of the non-technical issues related to EAM discussed previously may not impact the work of internal auditors in the same fashion as external auditors. Internal auditors generally have fewer restrictions placed on the scope of their work and are dedicated to a single organization. Client independence and legal liability are not necessarily relevant obstacles for internal auditors. How does the shift of continuous audit work from external auditors to internal auditors affect issues for EAM continuous auditing such as legal liability, access to systems, security over system change management, etc.? These questions raise several research challenges.

Challenge 6: Is the external auditor's willingness to rely on internal auditors' work affected by whether the evidence is gathered by human monitoring versus automated continuous auditing technologies?

Challenge 7: Are all of the independence issues mitigated when the external auditor is relying on the internal auditors' use of the continuous audit technologies rather than the external auditor using the system?

Challenge 8: How are the litigation issues altered when the potential audit failure occurs from the external auditor's reliance on the internal auditor's use of continuous audit technologies?

Opportunities for Research on Current Alternatives to Traditional EAM

While we present ghosting as an option for making EAM viable and as a solution to the drain on the efficiency of operational processing, we are unaware of any organization that has actually used this approach. An instantiation and testing of such an approach would provide substantial insight into the viability of such an approach, as well as insights into strategies for improvement and areas of difficulty that might be further assisted through research. To fully understand the issues that come from implementation in a large organization, it is likely that the greatest knowledge gains will come through partnering with an organization that would allow access to a ghosted version of their system. This may be difficult, but would still potentially be possible (see Jans et al. [2010] for example).

Challenge 9: Is ghosting a feasible (i.e. technically, operationally, financially) option for achieving EAM continuous audit objectives?

MCL certainly appears to be a feasible alternative to a live EAM implementation in certain instances. Still, MCL also poses challenges that would benefit from expanded research efforts. Similar to the financial transactional data obtained during the normal course of an audit retained in electronic working papers, the data extracted into the MCL application will need identical security measures and comfort provided to the client that the data is truly secure. Probably more significant, what would be the cost of retaining the vast amounts of data obtained from a firm's audit clients and is the cost worth the benefit? Additionally, while case studies such as that documented in Alles et al. (2006) and Alles et al. (2008) in their prototype development with Siemens provide insights, additional insights from broader scale implementations will provide even greater clarity on how such systems are and can be used and the challenges that come with them. For instance, KPMG's recent decision to license and deploy Caseware's IDEA system across all of their audits presents an excellent opportunity for studying and understanding how similar systems are adopted and utilized by audit teams. Related research could come from several theoretical perspectives to improve understanding of how these systems are used.

Challenge 10: How willing are auditors to accept continuous auditing applications as viable audit tools that support their decision making?

Challenge 11: How willing is management to accept auditors' use of continuous auditing applications? Is acceptance affected by whether the applications are targeted at managers' activities or their subordinates?

Adaptive structuration theory, for instance, might help in understanding how audit teams adjust systems usage to meet team objectives and team risk assessments (see Dowling 2009). Likewise, the widely used Technology Acceptance Model (TAM) may be of use in understanding acceptance of such technology by audit teams and the level of integration into the overall audit process. It should be recognized, however, that KPMG's usage of IDEA in the short term does not appear likely to include continuous auditing applications and that the insights gained now will provide only a partial view of how such technologies would be adopted and used for continuous auditing.

Another issue that the Siemens' implementation brings to the forefront is the problem in dealing with false positives and alarm floods, which affects all continuous auditing architectures

(Alles et al. 2006; Alles et al. 2008). If analytical criteria are set too stringently there is the potential to flag a large number of transactions as potentially erroneous or fraudulent and the information overload for the auditor could be substantial. Leaving criteria too loosely bounded, on the other hand, could fail to detect a large percentage of erroneous or fraudulent transactions. Thus, methodologies for addressing the inherent information overload are needed to improve the viability of continuous auditing. For instance, Perols and Murthy's (2009) work with information infusion may yield one such approach to mitigating the information load effects. Further examination of information infusion capabilities, as well as other alternative approaches, seem worthy of research attention. More sophisticated analytics that apply artificial intelligence techniques or other mathematical algorithms may also improve the detection capability of continuous auditing systems and to reduce the overload of false positives. Work such as that by Jans et al. (2010) suggests that such refined capabilities are possible with advanced analytic techniques.

Challenge 12: How do auditors (internal or external) adapt continuous audit applications to meet their audit strategies and approaches? How do auditors adapt the technology to cope with "alert flood"?

Challenge 13: What is the impact of "alert flood" on auditor information processing and decision making? What heuristics do auditors adopt in order to cope with "alert flood"? Is there a tipping point as to when "alert flood" overwhelms the auditor's ability to cope and adjust decision strategies?

Challenge 14: Can sophisticated analytics be developed that provide more refined testing under continuous audit applications? Can artificial intelligence techniques be used to improve continuous audit monitoring strategies?

Broad Research Questions

A number of research issues have been raised in the earlier sections that are critical to the evolution of continuous auditing, particularly within ERP system environments. Beyond those issues is a basic need to also consider the future from an integrative perspective that focuses on ERP development from a new systems design approach. In other words, if we were building ERP systems to function in today's environment, how might they look different? Would the existing EAM, EAM ghosting, and/or MCL continuous auditing architecture be effective for these ERPs?

Sutton (2006) broaches this subject in part through consideration of the changes in switching costs associated with new ERP system implementation in light of SOX requirements. In short, Sutton (2006) argues that a major additional cost has been added to the cost-benefit equation for switching ERP systems and to some degree even upgrading systems—the basic requirement that any new system will need to be fully documented for control processes and procedures with the potential costs perhaps mirroring first-year implementation cost when existing systems were documented. This cost provides increased incentive for companies not to switch ERP systems. There is also little incentive for non-adopters of ERP systems to adopt given the daunting tasks of implementing well-controlled systems and documenting those systems to meet SOX compliance requirements. How are these issues overcome?

The need is cyclical in some sense: adopters need methods for modeling effective internal controls within systems during the design and implementation phases and, subsequently, adopters need systems to better facilitate documentation of actual internal control systems that have resulted from implementations. One potential avenue for attacking this need is to consider how internal control structures can be integrated into modeling approaches used to design systems. McCarthy et al. (2005) is one such attempt to approach this issue through REA modeling. Their

approach provides a conceptual basis for thinking about how existing modeling approaches can be refined to improve the integration of internal control processes as an integral part of system and business process design.

The reverse side of this approach is the need for ERP systems that are essentially self-documenting. Development of modeling languages that incorporate internal control structures holds the best promise for developing such systems. In essence, if modeling approaches can be designed to drive design, then systems should similarly be able to take embedded models underlying the implementation and reverse out (i.e., automatically generate) documentation for the implemented structure. This could alleviate the documentation costs associated with SOX 404 that are a potential deterrent to implementing new systems (Sutton 2006). Extending this to self-analysis systems, the next logical step is to develop systems that can also self assess the underlying model to determine potential control weaknesses or deficiencies. Virsa's Control Calibrator, as originally designed, is an example of the potential to develop such systems, albeit on a limited scale, with its ability to assess an SAP implementation's enforcement of segregation of duties and its ability to identify weaknesses in segregation of duties (SAP 2008). This leaves a nested set of complex challenges for modelers willing to address the research needs.

Challenge 15: Can semantic modeling techniques, such as the REA model, effectively incorporate a comprehensive representation of internal control structures? Are such modeling techniques beneficial to strong control integration during systems implementations?

Challenge 16: Could the models generated through such modeling techniques as REA be used to automatically generate configurations of ERP systems, including which switches should be modified during implementation to tailor a system to business processes and the integration of appropriate control structures?

Challenge 17: Could ERP systems be designed to self-generate semantic models that represent the system's implementation and self-document control processes in place at a given point in time? Could continuous audit applications automatically adapt learn from such semantic models to configure the set of tests that should be used to monitor ERP processes?

As continuous auditing systems become increasingly prevalent, studies of adoption experiences will be critical to learning from early experiences. As noted in the previous section, studies of individual behavior patterns in terms of adopting and applying such systems will be important. Those behavior patterns should go beyond just the auditor side of the equation, however. We also need to understand the impact on auditees, including manager decision making. Hunton et al. (2008) provide preliminary evidence that manager's decision-making may be altered when they know they are being monitored. These behavioral effects should be carefully studied and assessed in considering the benefits and detriments that come with continuous auditing use.

Challenge 18: How are auditees' decision-making behaviors affected by continuous monitoring activities? Do personality factors moderate any such effect?

Challenge 19: How are auditees' perceptions of management's desired behavior impacted by the signaling effects of introducing continuous monitoring activities?

It should be noted here that such studies should also consider effects that exist at the organizational level. There is an emerging literature in the information systems domain that examines critical factors in the assimilation of new technologies—particularly strategic information systems (e.g., Chatterjee et al. 2002; Liang et al. 2007). The theories in this area hold strong potential for

helping to understand how well organizations assimilate and leverage continuous auditing systems. On the other hand, the unique nature of continuous auditing systems as a control mechanism may also yield advancements in existing theories of assimilation that expand the understanding of the applicability to a broader range of systems.

Challenge 20: What are the organizational factors that lead an organization to adopt continuous audit technologies?

Challenge 21: What are the organizational factors that drive strong assimilation of continuous audit technologies?

The research challenges put forth in this paper likely only scratch the surface. Our understanding of the impact of continuous auditing systems at this point in time is rudimentary and the opportunities for research that can have a significant impact are great.

IV. CONCLUSION

The development and pervasive use of ERP systems provides the critical infrastructure necessary for the effective evolution of the assurance function from a periodic event to an ongoing process through the integration of continuous auditing applications. Two principal system architectures for the development and continuous auditing applications have emerged in research literature. The Embedded Audit Module methodology developed early on in the domain's existence integrates continuous auditing functionality internally within the system of concern and operates in a truly real-time, continuous mode. Subsequently developed as an alternative to EAM, the Monitoring and Control Layer methodology focuses on an external module that interfaces with the system of concern to retrieve scheduled interfaces of selected data. Each of the continuous auditing approaches offers distinct advantages over the other.

Debreceny et al. (2005) examine the nature of EAM functionality and the relationship with ERP systems, perceived reasons for the lack of widespread adoption of EAM, and potential directions to increase the viability of the EAM approach to continuous auditing. In order to add additional clarity to the discussion, our study expands upon the limitations of EAM noted by Debreceny et al. (2005) and also explores the limitations of alternative continuous auditing approaches such as EAM ghosting and MCL. These limitations are examined from both technical and practical perspectives, as well as also examining the issues specifically from an audit perspective which carries major concerns of design and maintenance, client independence and auditor legal liability. We examined alternative strategies for implementation of continuous auditing, considering both (1) a modified EAM approach through the use of "ghosting" technology and (2) the MCL methodology. At present, MCL methodology appears to offer the most used alternative, as evidenced by both the recent increased attention in academic research (Alles et al. 2006; Kuhn and Sutton 2006; Alles et al. 2008) and the exclusive use of MCL for continuous auditing products developed by third-party vendors (AMR 2008).

As Debreceny et al. (2005) note, no prior research existed on EAM support within ERP systems at the time of their study. Only two additional research projects have surfaced since that point that examine continuous auditing in an ERP environment (i.e., Kuhn and Sutton 2006; Alles et al. 2006; Alles et al. 2008). Given the pervasiveness of ERP environments, additional research is necessary to advance the awareness, relevance, and practicality of ERP continuous auditing. In the course of discussing the issues fundamental to the use of EAM versus MCL, we have outlined a number of key research issues. The opportunities for research are plentiful and the need even greater when considered in light of wide ranging issues related to the potential demand for continuous monitoring and continuous auditing (e.g., Daigle and Lampe 2004, 2005), to the consideration of external auditor willingness and capability to implement and utilize continuous

auditing strategies, to the transference and increasing reliance of internal control work performed by internal auditors, to the need for additional instantiations of continuous auditing applications in "real" environments (e.g., Alles et al. 2006; Kuhn and Sutton 2006), and to the examination of innovative integrated IT platforms that cross technologies. Exploration has only just begun on understanding the behavioral effects on an organization's employees and managers (e.g., Hunton et al. 2008), but these issues are also critical to anticipating the impact as these systems become more widespread. Opportunities abound for research facilitating the efficient and effective implementation and utilization of continuous auditing capabilities.

REFERENCES

Alles, M., A. Kogan, and M. A. Vasarhelyi. 2002. Feasibility and economics of continuous assurance. *Auditing: A Journal of Practice & Theory* 21 (1): 125–138.

———, G. Brennan, A. Kogan, and M. A. Vasarhelyi. 2006. Continuous monitoring of business process controls: A pilot implementation of a continuous auditing system at Siemens. *International Journal of Accounting Information Systems* 7 (2): 137–161.

———, A. Kogan, and M. A. Vasarhelyi. 2008. Putting continuous auditing theory into practice: Lessons from two pilot implementations. *Journal of Information Systems* 22 (2): 195–214.

AMR Research, Inc. (AMR). 2008. Enterprise performance management: 2008 landscape series. Available at: http://www.approva.net/assets/resources/AMR_Research_REPORT_21828-The_Governance,_Risk_Management,_and_Com.pdf.

Behn, B. K., D. L. Searcy, and J. B. Woodroof. 2006. A within firm analysis of current and expected future audit lag determinants. *Journal of Information Systems* 20 (1): 65–86.

Braun, R. L., and H. E. Davis. 2003. Computer-assisted audit tools and techniques: analysis and perspectives. *Managerial Auditing Journal* 18 (9): 725–731.

Canadian Institute of Chartered Accountants and American Institute of Certified Public Accountants (CICA/AICPA). 1999. *Continuous Auditing. Research Report*. Toronto, Canada: CICA.

Chatterjee, D., R. Grewal, and V. Sambamurthy. 2002. Shaping up for e-commerce: Institutional enablers of the organizational assimilation of web technologies. *Management Information Systems Quarterly* 26 (2): 65–89.

Chen, H. H., S. H. Li, S. M. Huang, and Y. C. Hung. 2007. The development and performance evaluation of a continuous auditing assistance system. *International Journal of Electronic Finance* 1 (4): 460–472.

Daigle, R. J., and J. C. Lampe. 2004. The impact of the risk of consequence on the relative demand for continuous online assurance. *International Journal of Accounting Information Systems* 5 (3): 313–340.

———, and ———. 2005. The level of assurance precision and associated cost demanded when providing continuous online assurance in an environment open to assurance competition. *International Journal of Accounting Information Systems* 6 (2): 129–156.

Debreceny, R. S., G. L. Gray, W. L. Tham, K. Y. Goh, and P. L. Tang. 2003. The development of embedded audit modules to support continuous monitoring in the electronic commerce environment. *International Journal of Auditing* 7 (2): 169–185.

Debreceny, R. S., ———, J. Jun-Jin Ng, K. Siow-Ping Lee, and W. Yau. 2005. Embedded audit modules in enterprise resource planning systems: Implementation and functionality. *Journal of Information Systems* 19 (2): 7–27.

Deloitte. 2007. Security user segregation of duties—SAP GRC. Available at: http://www.Deloitte.com/dtt/case_study.

Dowling, C. 2009. Appropriate audit support system use: The influence of auditor, audit team and audit firm factors. *The Accounting Review*.

Ernst & Young. 2006. SEC and PCAOB roundtable discussion on implementation of internal control reporting provisions: Year Two. *Emerging Trends in Internal Controls* (May): 1–12.

Fogarty, J. 2005. *Continuous Auditing: Can it Become a Reality and What is its Impact on Auditing Standards*. Rutgers Tenth Continuous Auditing and Reporting Symposia. Newark, November.

Gerhardt, K. W. 2002. *The Emerging Role of the Information Systems Function in Modern Organizations:*

Technical and Managerial Challenges. The Peter Kiewitt Institute IST&E Research Round Table, Center for Management of Information Technology. Omaha, NE, October 25.

Greenbaum, J. 2008. SAP's business ByDesign: Explaining the delays. Available at: http://blogs.zdnet.com/Greenbaum/?p=165.

Groomer, S. M., and U. S. Murthy. 1989. Continuous auditing of database applications: An embedded audit module approach. *Journal of Information Systems* 3 (2): 53–69.

———, and ———. 2003. Monitoring high volume on-line transaction processing systems using a continuous sampling approach. *International Journal of Auditing* 7: 3–19.

Henrickson, R. 2009. *Practitioner Discussion of Principles and Problems of Audit Automation as a Precursor for Continuous Auditing*. University of Waterloo Centre for Information Integrity and Information Systems Assurance 6th Bi-Annual Research Symposium. Toronto, October.

Hunton, J. E., E. G. Mauldin, and P. R. Wheeler. 2008. Potential functional and dysfunctional effects of continuous monitoring. *The Accounting Review* 83 (6): 1551–1569.

Jans, M., N. Lybaert, and K. Vanhoof. 2010. Internal fraud risk reduction: Results of a data mining case study. *International Journal of Accounting Information Systems*.

Kogan, A., E. F. Sudit, and M. A. Vasarhelyi. 1999. Continuous online auditing: A program of research. *Journal of Information Systems* 13 (2): 87–103.

Kuhn, J. R., and S. G. Sutton. 2006. Learning from WorldCom: Implications for fraud detection through continuous assurance. *Journal of Emerging Technologies in Accounting* 3: 61–80.

Liang, H., N. Saraf, Q. Hu, and Y. Xue. 2007. Assimilation of enterprise systems: The effect of institutional pressures and the mediating role of top management. *Management Information Systems Quarterly* 31 (4): 59–88.

Loh, S., and R. Jamieson. 2008. *Continuous Assurance of EBusiness Transactions for Fraud Detection*. Sydney, Australia: University of New South Wales.

Martens, C. 2006. SAP to buy compliance software firm Virsa. *InfoWorld* (April 03).

McCarthy, W. E., G. Gal, and G. Geerts. 2005. *Semantic Specification and Automated Enforcement of Internal Control Procedures within Accounting Systems*. 10th World Continuous Auditing Conference, November.

Murthy, U. S. 2004. An analysis of the effects of continuous monitoring controls on e-commerce system performance. *Journal of Information Systems* 18 (2): 29–47.

Perols, J. L., and U. S. Murthy. 2009. *Information fusion in continuous assurance*. University of Waterloo Centre for Information Integrity and Information Systems Assurance 6th Bi-Annual Research Symposium, Toronto, October.

Reckers, P., S. Whitecotton, and J. Lowe. 2002. The effects of decision aid use and reliability on juror's evaluations of auditor liability. *The Accounting Review*: 77 (1) 185–202.

Rutgers Business School. 2005. Tenth continuous auditing and reporting symposia agenda. Available at: http://raw.rutgers.edu/10th/10thWCASAgendaV21nov12005URL.htm. University of Rutgers (Newark, November).

SAP. 2007. SAP Compliance Calibrator by Virsa Systems. Available at: http://www.SAP.com.

———. 2008. Stop security violations before they occur. Available at: http://www.sap.com/usa/solutions/business-suite/erp/financials/pdf/BWP_Stop_Security_Violations.pdf.

Sutton, S. G. 2006. *Implications of the U.S. Sarbanes-Oxley Act for Enterprise Systems Selection, Switching Costs, and Future Design*. Proceedings of the Third International Conference on Enterprise Systems and Accounting, Santorini, Greece.

———, R. Young, and P. McKenzie. 1995. An analysis of potential legal liability incurred through audit expert systems. *International Journal of Intelligent Systems in Accounting Finance & Management* 4 (3): 191–204.

Taylor, A. 2008. Preview of SAP business ByDesign. Available at: http://www.ondemandbeat.com/2008/02/27/preview-of-sap-business-bydesign/.

Vasarhelyi, M. A. 2006. Concepts in continuous assurance. Working paper, Rutgers University.

———, M. Alles, and A. Kogan. 2004. Principles of analytic monitoring for continuous assurance. *Journal of Emerging Technologies in Accounting* 1: 1–21.

———, and F. B. Halper. 1991. The continuous audit of online systems. *Auditing: A Journal of Practice & Theory* 10 (1): 110–125.

VMware. 2009. Best practice guidelines for SAP solutions on VMware infrastructure. Available at: http://www.vmware.com/files/pdf/whitepaper_SAP_bestpractice_jan08.pdf.

Woodroof, J., and D. Searcy. 2001. Continuous audit: Model development and implementation within a debt covenant compliance domain. *International Journal of Accounting Information Systems* 2 (3): 169–191.