Audit Automation for Implementing Continuous Auditing: Principles and Problems

Michael G. Alles
Alexander Kogan
Miklos A. Vasarhelyi

Rutgers Business School

Department of Accounting,
Business Ethics & Information Systems
180 University Ave
Newark, NJ 07102

Version: August 20, 2008[*]

Abstract: When implementing continuous auditing, experience indicates that auditors will likely attempt to first automate the processes that they already use, are comfortable with and are already accepted for external auditing and reporting purposes rather than trying to start from scratch, especially when dealing with audits of ongoing operations. However, because of the experience of low productivity and failed expectations with prior technology implementations that gave rise to the argument for business process reengineering in Hammer's (1990) article, "Don't Automate, Obliterate" special care needs to be taken when change is brought about by automation. As we argue in this paper, not only must audit automation be undertaken systematically, it also has to incorporate reengineering in the more limited sense of first transforming manual audit processes to facilitate their automation. This is not full blown reengineering of the clean sheet sort, but this hybrid approach is one that is more manageable—and marketable—from a change management perspective, and more likely to lead to a positive outcome. The key for avoiding the potential downsides of automation, though, is to have a clear understanding of what audit automation is trying to achieve and follow a methodical procedure to achieve those goals.

Keywords: audit automation, continuous auditing, internal audit, audit systems.

[*] Comments welcome. Please address them to kogan@rbsmail.rutgers.edu.

1. Introduction

Alles et al (2006) described a feasibility study undertaken by the IT Internal Audit department at Siemens Corporation, working with the authors, to create a continuous auditing (CA) system by automating the largely manual audit of its SAP systems. Since that initial study, the field of continuous auditing has rapidly developed, with vendors offering sophisticated IT products that facilitate CA implementation, as well as many other firms having begun to develop homegrown CA processes. This paper builds on the authors experience with such vendors and firms, as well work we undertook with Siemens on its second generation CA implementation. While the details of that particular case study is described elsewhere (Teeter and Brennan, 2008), in this paper we step back to draw general conclusions about the challenges that auditors will face when automating existing audit procedures for a CA environment, as well as the opportunities that they now have with new CA-enabling technologies.

As with Alles et al. (2006), our focus is on automation of an existing, predominantly manual audit process. While in years to come new CA audits may be created with a blank sheet approach, the experience of much technology implementation from mini-computers to ERP suggests that the change process is likely to be incremental rather than disruptive. Hence, as at Siemens, auditors will likely attempt to first automate the processes that they already use, are comfortable with and are already accepted for external auditing and reporting purposes rather than trying to start from scratch, especially when dealing with audits of ongoing operations. Moreover, audit standards have been largely written for a world in which technology may be an enabler, rather than the driver of audit processes as it is in CA, which again implies that the current need is for an understanding of automation as the primary mechanism used to bring about CA.

At the same time, precisely because of the experience of low productivity and failed expectations with prior technology implementations that gave rise to the argument for business process reengineering in Hammer's (1990) article, "Don't Automate, Obliterate" special care needs to be taken when change is brought about by automation. As we argue in this paper, not only must audit automation be undertaken systematically, it also has to incorporate reengineering in the more limited sense of first transforming manual audit processes to facilitate their automation. This is not full blown reengineering of the "throw

away the [manual] rule book" sort, but this hybrid approach is one that is more manageable—and marketable—from a change management perspective, and more likely to lead to a positive outcome. The key for avoiding the potential downsides of automation, though, is to have a clear understanding of what audit automation is trying to achieve and follow a methodical procedure to achieve those goals.

We begin by considering the case for audit automation and its relation to continuous auditing.

2. Drivers and Objectives of Audit Automation

Half a century has passed since the original utilization of computers in business. Even in those early days the tremendous potential impact of automation on accounting and auditing was understood and brought to the attention of the profession and academia (Keenoy, 1958). What was in the beginning a limited scope deployment focused on carefully selected business areas (such as payroll processing and inventory control) and utilized primarily by large enterprises, has since become ubiquitous for most business entities, and is now inseparable from doing business. Most business processes today are automated to various degrees, and businesses continue to invest in maintaining and expanding this automation through the acquisition of computer and telecommunication technologies and various enterprise systems, such as enterprise resource planning, data warehousing, supply chain management, and customer relationship management.

Automation of business processes has inevitably led to changes in auditing procedures and standards. Starting with the original release in 1973 of SAS 3 ("The Effects of EDP on the Auditor's Study and Evaluation of Internal Controls"), the utilization of modern information technology has made its way into the audit process, prompted by growing availability and decreasing cost of personal computing and office automation software (word processors and spreadsheets), as well as generic statistical software and dedicated computer-assisted audit techniques, such as Audit Command Language. These early deployments demonstrated the potential of IT in making labor-intensive repetitive audit work more efficient. The potential impact of audit automation on the changes in the audit process was originally researched almost quarter century ago by Vasarhelyi (1984). Recent empirical studies (Banker et al., 2002) have shown that IT-enabled audit automation indeed leads to significant productivity gains.

The importance of audit automation and the utilization of IT in modern audits has grown significantly in recent years due to both technological developments and changing regulatory environment (Janvrin et al., 2008). The passage of the Sarbanes-Oxley Act of 2002 (SOX), in particular, the requirements of its Section 404 greatly expanding the internal control work performed by the auditors, has resulted in a strong increase in demand for qualified audit personnel, leading to personnel shortages and audit cost increases. This creates an opportunity for automating audit processes to further increase their efficiency (Alles et al., 2006). While the problems facing public accounting firms with the cost and availability of qualified audit personnel have received much attention in the professional press, the budgetary pressure on internal audit departments post-SOX is probably even more severe. Thus, it comes as no surprise that many internal auditors are now at the forefront of deploying such modern audit automation technologies as continuous auditing.

Furthermore, the progressive sophistication of information technology underlying modern business processes has made the traditional approach of "audit around the computer" ineffective. For example, corporate enterprise resource planning (ERP) systems, designed for high volume online transaction processing, incorporate thousands of automated controls that can be configured in numerous ways. Manual verification of the status of these controls is becoming increasingly costly (Alles et al., 2006).

Additional drivers of audit automation adoption include the ever growing complexity of business transactions and increasing risk exposure of modern enterprises. This requires the audit to become more effective (Bedard et al., 2008). Several studies have shown that the deployment of IT and decision support systems to automate certain parts of the audit process usually results in a better controlled, higher quality audit (Manson et al., 1998; Dowling and Leech, 2007). Freeing human auditors from doing automatable audit work makes it possible for them to focus and spend more time on highly judgmental high risk areas (Vasarhelyi et al., 2004). It can also allow the auditors to increase the scope of the audit and utilize some additional audit procedures for higher coverage of various risk areas.

Automated audit procedures enable a dramatic increase in the scale of the audit since they are no longer limited by the constrained processing power of human beings, and thus they do not have to rely on sampling, and can process complete populations of transactions, another argument for the increased effectiveness of automated audit. Additionally, the

results of automated audit procedures are available in (close to) real time, thus increasing the timeliness of audit results.

Indeed, automation of audit procedures is both the necessary and sufficient condition for continuous auditing, and CA is a natural way of implementing an automated audit program. But while formalized audit procedures programmed into an automated audit system can be run continuously, they need not be, which is why we make a distinction in this paper between audit automation and continuous auditing. As Alles et al. (2002) argued, the frequency of auditing in a CA environment is more a function of the demand for information on an accelerated time frame, as opposed to the supply of the technology that makes more frequent assurance feasible. Thus, audit automation makes it possible for users to have "assurance on demand" as Alles et al. (2002) put it, but whether that demand is "continuous" in the everyday use of that term depends on the information that is being assured and the time frame of the decision problem that information is an input into. The bottom line, though, is that the prerequisite for the provision of any type of continuous assurance is an automated audit process.

Alles at al. (2006, 2008a, 2008b) break down continuous auditing into Continuous Control Monitoring (CCM) and Continuous Data Assurance (CDA):

CA = CCM + CDA

Examples of CCM include procedures for monitoring:

- Access Control and Authorizations
- System Configuration
- Business Process Settings

Examples of CDA include procedures for verifying:

- Master Data
- Transactions
- Key Process Metrics using analytics (including Continuity Equations)

While continuous monitoring of access controls and authorizations is well developed in computer security applications, monitoring enterprise system configuration and business process settings is an emerging area of development. Although our field work on audit

automation involved primarily CCM, the general principles and problems of audit automation generalized from that experience apply to both aspects of CA.

3. Audit Automation Change Management

The audit profession is inherently conservative given that its entire value added comes from the auditor's credible claims of objectivity and reliability. As a consequence, auditing processes, even more so than other business processes, have a tremendous amount of inertia. It follows that any audit automation project, as with any major change initiative in such circumstances, will have numerous barriers to change to overcome. This is why it is critical to ground audit automation projects in sound business process change methodologies developed in the management literature as a result of extensive experience with large scale IT implementations (see e.g., Davenport and Short, 1990; Wastell et al., 1994; Kettinger et al., 1997; Reijersa and Liman Mansar, 2005).

As research on ERP implementation success factors has convincingly shown (Umble et al., 2003; Botta-Genoulaz et al., 2005), for an automation project to even get launched, let alone succeed, senior executive champions have to take ownership of the project, both at the internal audit level, and at their reporting level in the C-suite or the audit committee. In the case of Siemens the champion that brought the authors on board was the then head of the IT Internal Audit Department. The fact that we are increasingly coming across executives at firms with titles such as "Associate Director, Continuous Assurance" (in the case of BD Corporation) indicates that such champions are becoming institutionalized in firms as CA goes mainstream.

The first critical task of audit automation champions will be to identify and engage project stakeholders. In addition to internal auditors, these stakeholders will include business process owners and IT personnel. Again, the use of such multifunctional teams is a standard recommendation of change management theory, but in the case of audit automation the problem is compounded by the need of internal audit to be aware of the needs of the external auditor, while also balancing the demands of the IT process owners and line managers. The composition of audit automation teams must reflect the multi-faceted nature of the task at hand.

The reason for having a high powered team with a senior level champion is obvious when considering the complexity inherent in automating audit processes initially designed to be done largely manually. In our experience, even very experienced auditors differ in how such procedures are carried out in practice, which translates into differences in how to transform the process into an automated one, what the objective of the process should be and how much weight should be placed on a particular process or on a possible compensating control.

A powerful way of increasing the quality and reliability of audit automation results would be to diversify risk and bring out differences by utilizing duplicate audit automation teams, and then comparing the resulting automated audit programs. Resolving the inevitable disagreements between the duplicate teams would greatly improve the final automated program. While this approach is often utilized in academic research (e.g., to make sure that human responses are coded properly), we have yet to come across any instance in which such a procedure has been adopted. It is simply too expensive, both in terms of human resources and time to be feasible for the vast majority of enterprises. Therefore, alternative measures have to be utilized to assure the quality of the automated audit program both during the automation process and to verify the completed product.

One such alternative is for the automated audit procedure developed by the automation team to be verified independently by experienced auditors who took no part in developing it. A vitally important check on the audit automation process is the need to satisfy the external auditor, and to retain their reliance on the internal audit process. As, in accordance with SAS 65 ("The Auditor's Consideration of the Internal Audit Function in an Audit of Financial Statements"), the external auditor evaluated the quality and effectiveness of the original manual audit process, such evaluation can encompass the automation process, the finished automated audit program, or ideally, both. In either case, demonstrating that the automation team followed a systematic procedure is an essential element in satisfying the external auditor.

4. Formalizing and Reengineering the Audit Program

As Alles et al. (2006) indicated, before audit procedures can be automated, they must first be formalized: *"Automation requires formalization of audit procedures. Approved audit programs are not*

highly formalized and most often reflect the legacy of the traditional manual audit/interview approach to auditing. Different human auditors interpret the same program somewhat differently. Our pilot study analysis of the approved internal IT audit program shows that certain parts of the program are formalizable while other parts are not."

Indeed, since the audit programs are designed by human auditors for execution by human auditors who are presumed to largely share their own knowledge and judgment, audit procedures in these programs are not completely formal and as such, they leave open significant room for interpretation. This is extremely problematic for the audit automation process, though, since, as confirmed by experience, even highly qualified human auditors will often disagree about the precise interpretation of a particular procedure. Whether this results in uneven audit quality is an empirical issue, and one outside the scope of this paper. What is undeniable though is that the resulting lack of consistency is one of the key barriers towards audit automation.

While formalization is a prerequisite of automation, formalizing an audit program has wide ranging benefits not limited to automation. By eliminating possible inconsistencies in program interpretation, the scope, scale and exact nature of audit procedures will be assured. Consequently, it can lead to the improved quality of results, and increased confidence in the audit as a whole, as was previously found to be the case after limited scope audit automation projects (Manson et al., 1998; Dowling and Leech, 2007). It should also decrease long-run audit costs due to the elimination of the on-going labor-intensive task of interpreting an ambiguous audit program. Additionally, it will drastically simplify and improve training of new auditors.

Many will argue that an audit process should not be formalized because of the need to retain the flexibility to interpret it suitably in differing future circumstances. The counterargument to that is to better specify what such circumstances of concern are and to systematically develop formal procedures to deal with them when they arise, as opposed to risking audit failure by building in excessive flexibility. Indeed, in our experience, auditors would simply leave out entire parts of the required audit manual by stating something like "well, I know that this was only intended to apply to our operations in China and so it is not relevant at this site". While it may be acceptable for a very senior and highly experienced lead auditor to make such a judgment, what happens when the audit is carried out by someone less

qualified, as will inevitably occur at some point due to resource constraints? The purpose of audit automation is to have areas of flexibility planned for rather than inserted haphazardly. As we argue below, baselining is a formal procedure that can be used to capture areas where the audit process needs to evolve.

Formalizing an audit program is a difficult endeavor. It can be very laborious and costly because a formal procedure has to be very specific and detailed, and it has to describe the precise modifications to be used in various conditions. This problem is compounded by the difficulties that many humans (even properly educated and trained ones) experience with logical reasoning and formal thinking (see e.g., Holvikivi, 2007; Rittgen, 2000). To address this problem, the audit automation project can utilize the methodology of knowledge engineering, especially knowledge elicitation, developed originally for expert systems and further enhanced as those evolved into modern knowledge-based systems (see, e.g., Cooke, 1994; Hoffman, 1995; Studer et al., 1998; Ford and Sterman, 1998; Schreiber et al., 2000; Kamsu Foguem et al., 2008).

Since manual audit programs were not designed for automation, formalizable and judgmental procedures are often intermixed. To formalize and automate such a program, a redesign is usually required to separate out formalizable and automatable audit procedures from the others. Such a redesign amounts to reengineering the audit program and should be done systematically (as opposed to ad-hock) and based on the top-down analysis of enterprise risks (Bell et al., 1997) to make sure that the redesigned procedures appropriately address all exposure areas.

The objective of reengineering is not only to enable automation by separating out the formalized audit procedures, but, more significantly, to maximize the proportion of automatable procedures in the audit program, and thus to reduce the reliance of audit procedures on informal judgmental techniques. An additional argument in favor of increasing the proportion of automated procedures in a reengineered audit program is due to the fact that these automated procedures can be performed much more frequently than the eliminated manual methods they substitute for.

Not everything can be made completely formal. Certain complex judgments are not amenable to formalization. Formalization is particularly difficult (if not impossible) whenever audit procedures have to deal with the analysis of modern complex business

contracts. At the same time, the possibility of formalization is often underestimated, and when an earnest effort is made to formalize audit procedures, the results often exceed the most optimistic expectations. Alles et al. (2006) concluded this much in their study of Siemens first generation CA implementation, but importantly, Teeter and Brennan (2008) indicated that recent advances in CA-enabling software, such as Approva, have increased the scope for formalization and audit automation.

Siemens' internal audit methodology for SAP facilities involves the carrying out of several hundred of "audit action sheets" by internal auditors at the auditee site. Alles et al. (2006) indicated that about 25% of the audit actions could be fully automated due to their deterministic nature. But Teeter and Brennan (2008), in their study of the CA initiative based on an Approva system foundation concluded that about 68% of the actions could be automated to some extent. Considering that some of these automated steps would be performed in a daily monitoring mode (as opposed to the 18 to 24 month cycle of SAP audits) the strength of its evidence would be much stronger and conceivably could replace much of the residual 32% non-automated evidence. Such replacements have been shown in some past experience to be the main benefit of audit automation (Fischer, 1996).

5. Baseline Monitoring ("Baselining")

Baseline monitoring or "baselining" for short is a well established procedure in configuration management and IT security, defining the "what should be" state that is used as a benchmark against which the current state can be compared. As applied to enterprise systems, baseline is defined as a set of system configuration and business process settings at a given, reference point of time.

Baselining is facilitated in audit automation through the use of the "snapshot" feature of audit automation software (such as Approva), which does precisely what the name suggests (albeit only for the SAP tables that the program monitors). The use of baselining allows the automation of audit procedures that would be difficult to formalize explicitly, by allowing for a simple before/after comparison. The price to pay for this approach towards automation is the extensive manual effort to initially verify all the values in the baseline to make sure that they are appropriate. This verification relies on human judgment typically supported by manual techniques such as interviewing, investigations and observations. However,

subsequent verification of the appropriateness of enterprise system configuration and business process settings can be completely automated since it is now reduced to the comparison of the current values to the values in the baseline. Any deviations from the baseline become exceptions that trigger alarms to be analyzed by auditors.

The effectiveness and efficiency of baseline monitoring depends on a number of critical issues. The first one is the definition of the baseline, i.e., which parameters should be included in it. If a certain parameter changes frequently in the process of routine business operations, then its inclusion in the baseline may be inappropriate if auditor's manual verification of every such change is not justified by the risk-cost tradeoff. Generally, the more static the parameters are, the better they are suitable for baselining.

Ongoing business operations will inevitably lead to changes resulting in alarms analyzed by auditors. If the analysis confirms the validity of a change, then, potentially, the verified value can be viewed as cleared for adding to the baseline. On the other hand, given the critical importance of the baseline and its security, it can be more prudent to accept this value only temporarily by deactivating the triggering of this particular alarm, and keep accumulating the deltas until some later time when additional verification of changes can be performed before redefining the baseline. That time will be an appropriate moment to analyze the experience of baseline monitoring with the objective of updating its definition – eliminating certain parameters that turned out to be too volatile, and/or adding some other parameters found to be critical after the initial definition of the baseline.

Indeed, such alarms should be of great interest to auditors if their analysis indicates that over time, once previously stable baseline parameters begin to show a consistent pattern of volatility. That would be an indication of the need to revisit the audit process and its underlying operational assumptions, giving a built in change management process to the audit automation program. Establishing a boundary as a benchmark and then monitoring exceptions as an indicator of required change is the basis for the model of strategy as arising from control proposed by Robert Simons (1995a, 1995b).

Alles and Datar (2004) used that framework when developing a management-control perspective on financial accounting standard setting and the SOX Section 404 requirements. Simons's "Levers of Control" framework consists of four complementary types of controls: boundary controls, belief controls, diagnostic controls and interactive controls. As Alles and

Datar (2004) state: *"Interactive controls are particularly important in control theory because it is they that tell senior managers when they need to change strategy—and so, when they have to alter the belief, boundary and diagnostic controls that put that strategy into practice. In other words, interactive controls make the entire control architecture dynamic, enabling it to evolve as underlying conditions change, or as core assumptions are proved invalid."*

The application to baselining is to draw the analogy of the baseline to a boundary control and to develop an analytic engine for alarms as an interactive control that would indicate when exceptions are being caused by a change in the underlying operations, thus necessitating an evolution of the automated audit, as opposed to an anomaly caused by inappropriate behavior by the auditee. In other words, baselining and their technological enabler, the "snapshot" feature of monitoring software, are simply tools, means towards an end. Putting them to work as an automated audit procedure requires that they be overlaid with an effective control framework that will enable their transformation into a dynamic monitoring process. It is possible that combining Simons' framework with the very powerful snapshot capabilities of modern audit enabling software will lead to the development of a highly capable CA system that goes beyond first-generation audit automation.

However, it needs to be reiterated that the initial manual verification of baseline values is a critical stage of audit automation. Any mistake made at this stage will be leveraged since the system will automatically perpetuate the mistake indefinitely. This is an indication of the different set of risks that can arise in automated systems. While in manual auditing there is always a chance that a mistake made during a particular audit cycle can be corrected during subsequent periods, in baselining there is only one chance during the initial verification stage to get things right.

Another critically important issue is the security of baseline—both in its definition and its current values. If one can compromise the security of the baseline and manipulate the definition of the baseline, say by removing from it certain parameters or by changing certain values in the baseline, then one can potentially open gaping holes in the system without anybody ever noticing. Thus, securing the baseline must be a well-developed feature of an automated auditing system.

6. Architecture of Automated Auditing

After the creation of an automated audit program, it has to be implemented in audit software. This software can be categorized along its following three dimensions: 1. Structure, 2. Access, and 3. Platform.

In terms of structure, audit software can be either integrated or distributed. It is natural to mimic the structure of the enterprise software being audited: if it is tightly integrated, the auditing software can be a tightly integrated system as well, while in the case of loosely coupled enterprise applications, a distributed system consisting of multiple auditing software agents will be a better fit.

Auditing software's access to the enterprise system and data can be either direct or intermediated. As the name direct suggests, in this case auditing software has access to the enterprise system implementing the business processes and containing source data being audited. Depending on the type of the enterprise system, this interaction can be either with its database or the application layer. If the direct access is too cumbersome, expensive, or infeasible to set up, then intermediated access is in order, typically through a business data warehouse. This approached is usually the only option in the case of highly heterogeneous loosely coupled legacy enterprise system landscapes.

The platform of automated audit software can be either common with the enterprise system, or completely separate. If the common enterprise platform hosts the audit software, the latter is usually referred to as an embedded audit module (EAM). Enterprise software vendors are naturally positioned to provide such software, even though until very recently they provided only rudimentary capabilities (Debreceny et al., 2005). If the audit software is hosted on a separate platform, it is usually referred to as monitoring and control layer (MCL), and this type of audit software is typically provided by third party vendors and audit firms.

While EAMs are usually permanently installed on the enterprise platform, one can also utilize an automated audit software architecture based on mobile code. In this architecture, the code implementing certain automated audit procedures is transported over the network to the enterprise platform on an as needed basis to execute its procedures there, and the code remains there for as long as needed. The primary reasons for executing audit procedures (whether in the form of EAMs or mobile agents) on the common enterprise platform are following.

First, it protects against network connectivity outages. Since remote code critically relies on the availability of connection to the enterprise system for access, it will be effectively disabled if the connectivity is lost (whether accidentally or intentionally). While modern networks are getting increasingly more reliable, sporadic connectivity outages still present a significant problem. On the other hand, resident code is obviously completely immune to this problem.

Second, the execution of resident code can be triggered by events in the enterprise system, while remote procedures can execute only after they retrieve information at a scheduled time. Event-triggered execution of audit procedures potentially reduces their latency to zero. Additionally, their latency is not affected by possible network congestion, which can significantly increase the latency of remote procedures.

Third, it is usually more efficient to process large volumes of enterprise data on site as compared with moving that data over the network for remote processing. The tradeoff here will depend on the processing capabilities of the enterprise system and on its load at the moment when processing is needed.

While the benefits described above seem to provide strong support for basing the architecture of automated audit on EAMs or mobile agents, there are extremely difficult problems associated with relying on the enterprise system for audit code execution.

On the one hand, there is legitimate concern on the part of the enterprise platform owner about the possibly adverse impact of the auditing code on the enterprise system itself. This impact can be caused by simply imposing a taxing computational load that can lead to the degradation of response time of routine enterprise transaction processing. To mitigate this issue, the enterprise platform can limit the amount of processing it provides to the auditing code, thus somewhat limiting its abilities. An even more serious concern on the part of the enterprise system owners is the possible interference by the code (either accidental or malicious) in the workings of the enterprise system. This is the reason for protecting the enterprise platform against a (possibly malicious) EAM or mobile agent. Modern IT provides well developed facilities for dealing with this problem in the form of a strictly controlled execution environment such as a sand box or a virtual machine.

The other side of the issues discussed above is the necessity to protect the EAM or mobile agent auditing code against possible manipulation by the enterprise platform. Given that the superuser privileges for the enterprise system are held by the enterprise IT personnel, the integrity of the audit code processing is always in question since it is the objective of this code to check on the enterprise system and its personnel. This problem has been discussed in the literature under the name of the *malicious host problem* (Jansen and Karygiannis, 1999; Claessens et al., 2003), and it is considered to be extremely difficult (if not infeasible). While there have been proposed numerous ways of dealing with it (see e.g., Stengel et al, 2005; Futoransky et al, 2006; Shao and Zhou, 2006; Topaloglu and Bayrak, 2008), and there are some quite complex ways of detecting the problem, no solution has been universally accepted as being able to prevent the host from interfering with code's execution.

The extreme difficulty (if not impossibility) of protecting the EAM or mobile agent auditing code from possible manipulation by the enterprise platform puts in question the integrity of results provided by this auditing code. This lack of trust in the audit results outweighs the benefits of the resident code described above, and serves as one of the critical reasons for basing automated auditing architecture on remote monitoring of enterprise systems.

7. Handling, Evaluation and Integration of Audit Evidence

While it is the ultimate objective of any business enterprise to have a totally reliable control system that will not have any exceptional events or anomalous situations, this ideal is never achieved, and the auditing system will be generating alarms caused by anomalies and exceptions. These alarms will be delivered by automatic means (e-mail, instant and wireless messaging) to the appropriate auditors and enterprise personnel responsible for resolving them. While the automated auditing system keeps track of each event, it is essential to have an automated closed loop process for capturing information about the corrective actions and assuring that these actions resolve the underlying problem.

As the results of resolving exceptional events and anomalous situations identify various control failures of the enterprise system, the auditing system should have a built-in mechanism for evaluating how significant these failures are, and making these evaluations available in a timely manner to the relevant stakeholders (auditors and upper management). To make such automatic evaluations possible, the procedures in the automated auditing

system have to be organized in accordance with the enterprise risk model to associate appropriate risks to various control failures.

While individual evaluations of control failures are no doubt important, large enterprises in particular would be interested in aggregating audit evidence to see the "big picture" of current enterprise exposure. The development of sound theoretical methodology for measuring internal control performance and aggregating audit evidence that would be practically applicable in modern large enterprises presents serious challenges. Theoretical studies of internal controls design and evaluation undertaken over the years (Cushing, 1974; Srinidhi, 1988; Vasarhelyi and Srinidhi, 1989; Knorr and Stormer, 2001) can provide a foundation for future practical developments which are yet to come. In the meantime, various ad hoc solutions and simplifying assumptions can be utilized to build a continuous auditing dashboard that in real time provides an aggregate view of enterprise control problems.

8. Software for Audit Automation

While it is certainly possible to design, develop and implement a custom-made automated auditing system in house, the expense and expertise requirements of such a project make it prohibitively expensive, if not outright infeasible, for the vast majority of cases. It is therefore not surprising that there is an emerging industry of packaged software developed to support audit automation or at least some of its aspects.

A convenient way of categorizing the current software offerings is in accordance with the breakdown of CA as consisting of CCM and CDA. While the vendors are attempting to integrate in their packages as many features as possible, they still typically exhibit strength in one of the two components. The well-established CAATs vendors ACL and CaseWare IDEA have extended their products to position them as continuous monitoring solutions. ACL in particular has invested significant efforts into providing what they call "continuous controls monitoring" solutions. Despite the name, in the terminology of this paper these solutions should be categorized as CDA since the substance of their tests is transaction verification and analysis focused on making inference about the functioning of controls (as opposed to direct tests of controls through monitoring of their settings). A relative

newcomer to this area is Oversight Systems which also focuses on CDA and puts emphasis on providing hosted monitoring solutions.

The common feature of CDA offerings is their utilization of their internal common data models to which enterprise data is mapped by the extract, transfer and load (ETL) subroutines. This system architecture allows for a relatively easy accommodation of many different enterprise systems (or even home-grown solutions) through the development of additional ETL modules to accommodate additional systems. The test libraries and the main processing subroutines usually do not have to be changed.

While the common data model architecture is utilized successfully in CDA solutions, the systems that implement CCM directly do not use it. The reason is the great diversity of business process automation in enterprise systems. The very significant differences in the types of business objects, process configurations and controls seem to make the common model too complex to be cost-effectively designed and implemented in CCM solutions. This is why these solutions develop special CCM subroutines targeted at specific enterprise systems. Not surprisingly, the two pioneering offerings in this field – Approva and VIRSA – were targeted at SAP R/3 (recently known as mySAP ECC). Approva has since extended its offerings to target other ERP systems, most notably Oracle E-Business Suite. Such extensions are quite laborious since they require the reimplementation of the CCM test libraries and processing for each new enterprise system. On the other hand, VIRSA has since been acquired by SAP itself, and has become the core of the SAP's governance, risk and compliance (GRC) offering. To keep up in its competition with SAP, Oracle acquired in the fall of 2007 a major GRC and CCM vendor LogicalApps, whose offerings were naturally targeted at the Oracle E-Business Suite.

The area of GRC is still maturing and has a very large number of vendors, many of them small, though some major vendors do have a presence, such as IBM, with its Workplace for Business Controls and Reporting. Among other notable offerings in this market one can find Paisley Enterprise GRC, OpenPages, AXENTIS Enterprise, BWise, and Protiviti Governance Portal. Many of the solutions in this market are not much more than customized document management systems with GRC-specific templates, though there is a pronounced trend to enhance these offerings with automatic control testing and monitoring

functionality that would bring these solutions closer to the fully developed CCM and/or CDA systems.

9. Scalability of Audit Automation

Automating a manual audit program requires a significant startup expense. This fixed cost may become a significant hurdle in the way of audit automation if an enterprise has no way of amortizing this cost over different enterprise units, since automation of highly specific audit procedures for different enterprise units can incur prohibitive costs. Automation will be scalable across the enterprise only if the repetitive audit procedure automation costs are eliminated.

There are a number of strategies for making audit automation scalable. The most immediate one is the parameterization of automated audit procedures. Given the expected significant homogeneity of enterprise business processes, it is likely that one can make automated audit procedures sufficiently generic by introducing various parameters describing systems, processes and business artifacts. Then the implementation of automated audit in additional enterprise units can be reduced to properly configuring the already developed system by assigning the appropriate parameter values.

In addition to parameterization, scalability of the audit program can be enhanced through hierarchical structuring of automated audit procedures – from the most generic audit procedures applicable across the enterprise to the more specific ones for major units and subunits. The feasibility of such structuring is enabled by the natural hierarchy of business enterprises and the risk-based top-down approach towards audit program development. This will also facilitate audit program maintenance through hierarchical updates: given a change in the processes at a certain node of the enterprise hierarchy, only the audit procedures in the hierarchical sub-tree rooted at this node will have to be reviewed and, possibly, revised.

9. Securing Continuous Auditing

Based on the analysis above, it is likely that an automated auditing system will be implemented as a MCL hosted on its own platform. To assure the integrity of its results, this system must be thoroughly secured. One of the issues that critically affect this system security is the control of the continuous auditing software, and its associated hardware,

which can be either under the authority of the auditee's IT department, or under the internal auditors themselves. Although the latter is intrinsically more secure, there are numerous practical matters that can favor the former. While we have come across instances in which the internal auditors have their own CA software (albeit, running on the firm's general IT systems), the cost and complexity of software such as Approva makes it far more likely to be entirely run by the firm's IT department (or outsourced to the vendors or third parties) with access provided to the auditors. In this case, one has to pay particular attention to access security.

Logical access security of the auditing system is even more critical since any compromise of the system can potentially be used to cover up for undesirable events in the enterprise system. As Alles et al. (2002, 2004) argued, automated audit systems are in one sense more vulnerable than manual systems due to the lack of "another-pair-of-eyes" controls and the leveraging of a mistake—intentional or otherwise—every time the CA system runs. Maintaining logical security is particularly problematic since it requires advanced system management IT skills which may not be easily available in internal auditing. The super-user privileges in the auditing system are figuratively speaking the "keys to the kingdom", and their safeguarding requires utmost attention, which is why this is a key control for SOX 404 certification.

To mitigate this exposure, comprehensive logging of all super-user activities should also be implemented and constantly monitored by the internal auditors, as Alles et al. (2004) proposed. An important control over the security of the auditing system consists in exporting its settings and cryptographic check-sums of its code to an external storage facility or/and non-volatile storage medium. Then these setting can be imported back into the CA system as needed, and the cryptographic check-sums of the system code can be recalculated to verify the integrity of the CA system. This is obviously a de-facto tertiary monitoring process, and the administration of this process should be done by dedicated internal audit personnel.

10. Concluding Remarks

The practice of audit automation will be strongly influenced by the ongoing software development and maturing of the field of GRC. AMR Research projects that spending on

government, risk and compliance applications and services will top $32.1 billion in 2008, up 7.4 % from 2007.[1] In 2009, growth is projected at 7 %. These are very significant business expenditures, and the competition among the software vendors is likely to be fierce. Future functionality developments in GRC are likely to be accompanied by major consolidations in this so far very fragmented market.

As more of the smaller companies start deploying GRC and implementing elements of audit automation in their internal audit departments, ongoing IT maintenance requirements and the lack of qualified IT personnel are likely to lead to the growing popularity of hosted or on-demand solutions. This approach my also alleviate the serious independence issues associated with reliance on the internal MIS department personnel for auditing system management.

The requirements of supporting the continuous operation of automated audit procedures by properly documenting their results and keeping this documentation up-to-date on a continuous basis is likely to lead in the long run to the integration of audit automation systems with audit working papers software. While there is no evidence yet in the marketplace of this integration, the amount of manual work associated with maintaining such documentation will make any audit automation project essentially incomplete without it.

The ongoing development and implementation of automation in auditing will likely lead to major transformation of internal audit. This transformation will involve the structure of the departments, the skill sets of internal auditors, and the role that the internal audit departments play in the organization, especially with respect to their interaction with the other units involved in the managerial control over the enterprise. The perennial issue of how to delimit the responsibilities of internal auditors and under what circumstances (if ever) their intervention in the enterprise process should be allowed will become even more acute.

Since audit automation will allow internal auditors to obtain high quality audit evidence about the operations of the enterprise on a continuous basis, it should make it possible for external auditors to rely more on their work. Such reliance, of course, will have to be accompanied by the appropriate arrangements to assure the external auditors about the

[1] http://www.financialweek.com/apps/pbcs.dll/article?AID=/20080428/REG/490727472

quality of work the automated auditing systems are providing. It is likely that it will also result in significant changes in the nature of audit procedures performed by the external auditor, and these changes in turn may lead to structural changes in audit engagements, and may, with time, reshape the external audit as we know it today.

References

1. Alles, M.A., and Datar, S. 2004. Cooking the Books: A management-control perspective on financial accounting standard setting and the section 404 requirements of the Sarbanes-Oxley Act. *International Journal of Disclosure and Governance,* Vol. 1, No. 2, 119–137.
2. Alles, M.G., A. Kogan, M.A. Vasarhelyi. 2002. Feasibility and economics of continuous assurance. *Auditing: A Journal of Practice and Theory*, Vol. 21, No. 1 (March), 125–138.
3. Alles, M.G., A. Kogan, M.A. Vasarhelyi. 2004. Restoring Auditor Credibility: Tertiary Monitoring and Logging of Continuous Assurance Systems. *International Journal of Accounting Information Systems*, Vol. 5, No. 2 (June), 183–202.
4. Alles, M.G., G. Brennan, A. Kogan, 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*, Vol.7, 137–161.
5. Alles, M.G., A. Kogan, M.A. Vasarhelyi, J. Wu. 2008a. *Continuous Data Level Auditing Using Continuity Equations*. Unpublished working paper, Rutgers Business School.
6. Alles, M.G., A. Kogan, M.A. Vasarhelyi. 2008b. Putting continuous auditing theory into practice: Lessons from two pilot implementations. *Journal of Information Systems*, forthcoming.
7. Banker, R. D., H. Chang, and Y. Kao. 2002. Impact of information technology on public accounting firm productivity. *Journal of Information Systems* 16 (2): 209–222.
8. Bedard, J. C., D. R. Deis, M. B. Curtis, J. G. Jenkins. 2008. Risk monitoring and control in audit firms: A research synthesis. *Auditing: A Journal of Practice & Theory*, Vol. 27, No. 1 (May), 187–218.
9. Bell, T.B., F. Marrs, I.Solomon, H. Thomas. 1997. *Auditing Organizations Through a Strategic-Systems Lens: The KPMG Business Measurement Process.* KPMG, Montvale, NJ.
10. Botta-Genoulaz, V., P.-A. Millet, B. Grabot. 2005. A survey on the recent research literature on ERP systems. *Computers in Industry*, Vol. 56, No. 6 (Aug.), 510–522.
11. Claessens, J., B. Preneel, J. Vandewalle. 2003. (How) can mobile agents do secure electronic transactions on untrusted hosts? A survey of the security issues and the current solutions. *ACM Transactions on Internet Technology*, Vol. 3, No. 1 (February), 28–48.
12. Cooke, N. J. 1994. Varieties of knowledge elicitation techniques. International *Journal of Human-Computer Studies*, Vol. 41, No. 6 (December), 801–849.

13. Cushing, B. E. 1974. A Mathematical Approach to the Analysis and Design of Internal Control Systems. *The Accounting Review*, Vol. 49, No. 1 (Jan.), 24–41.

14. Davenport, T.H. and J.E. Short. 1990. The new industrial engineering: Information technology and business process redesign, *Sloan Management Review* (Summer): 11-27.

15. Debreceny, R. S., G. L. Gray, J. J.-J. Ng, K. S.-P. Lee, W.-F. Yau. 2005. Embedded audit modules in Enterprise Resource Planning Systems: Implementation and functionality. *Journal of Information Systems*, Vol. 19, No. 2 (Fall), 7–27.

16. Dowling, C. and S. Leech. 2007. Audit support systems and decision aids: Current practice and opportunities for future research. *International Journal of Accounting Information Systems* 8: 92–116.

17. Fischer M.J. 1996. "Real-izing" the benefits of new technologies as a source of audit evidence: An interpretive field study. *Accounting, Organizations and Society*, Vol. 21, No. 2/3, (Feb.-Apr.), 219–242.

18. Ford, D. and J. Sterman. 1998. Expert knowledge elicitation for improving mental and formal models. *System Dynamics Review*, 14(4), 309–340.

19. Futoransky, A., E. Kargieman, C. Sarraute, A. Waissbein. 2006. Foundations and applications for secure triggers. *ACM Transactions on Information and System Security*, Vol. 9, No. 1 (February), 94–112.

20. Hammer, M. 1990. Reengineering work: Don't automate, obliterate. *Harvard Business Review*, July-August, 104–112.

21. Hoffman, R.R., N.R. Shadbolt, A.M. Burton, G. Klein. 1995. Eliciting knowledge from experts: A methodological analysis. *Organizational Behavior and Human Decision Processes*, Vol. 62, No. 2 (May), 129–158.

22. Holvikivi, J. 2007. Logical reasoning ability in engineering students: A case study. *IEEE Transactions on Education*, Vol. 50, No. 4 (Nov.), 367–372.

23. Jansen, W. and T. Karygiannis. 1999. Mobile agent security. *NIST Special Publication* 800-19 (October). http://csrc.nist.gov/publications/nistpubs/800-19/sp800-19.pdf

24. Janvrin, D., J. Bierstaker, D. J. Lowe. 2008. An Examination of Audit Information Technology Use and Perceived Importance. *Accounting Horizons*, Vol. 22, No. 1 (March), 1–21.

25. Kamsu Foguem, B., T. Coudert, C. Béler, L. Geneste. 2008. Knowledge formalization in experience feedback processes: An ontology-based approach. *Computers in Industry* 59, 694–710.

26. Keenoy, C. L. 1958. The impact of automation on the field of accounting. *The Accounting Review*, Vol. 33, No. 2, (April), 230–236.

27. Kettinger, W. J., J.T.C. Teng, S. Guha. 1997. Business process change: A study of methodologies, techniques, and tools. *MIS Quarterly*, Vol. 21, No. 1 (Mar.), 55–80.

28. Knorr, K. and H. Stormer. 2001. Modeling and analyzing separation of duties in workflow environments. *Proceedings of the IFIP TC11 Sixteenth Annual Working Conference on Information Security*, 199–212.

29. Manson, S., S. McCartney, M. Sherer, W. A. Wallace. 1998. Audit automation in the UK and the US: A comparative study. *International Journal of Auditing*, Vol. 2, No. 3 (Nov.), 233–246.

30. Reijersa, H.A. and S. Liman Mansar. 2005. Best practices in business process redesign: An overview and qualitative evaluation of successful redesign heuristics. *Omega* 33, 283–306.

31. Rittgen P.. 2000. Paving the road to business process automation. *Proceedings of the 8th European Conference on Information Systems (ECIS)*. http://is2.lse.ac.uk/asp/aspecis/20000050.pdf

32. G. Schreiber, H. Akkermans, A. Anjewierden, R.d. Hoog, N. Shadbolt, W.v.d. Velde, B. Wielinga. 2000. *Knowledge Engineering and Management*, MIT Press, Cambridge, MA.

33. Shao, M.-H. and J. Zhou. 2006. Protecting mobile-agent data collection against blocking attacks. *Computer Standards & Interfaces* **28**, 600–611.

34. Simons, R. 1995a. Control in an age of empowerment. *Harvard Business Review*, March–April, 80–88.

35. Simons, R. 1995b. *Levers of Control: How Managers Use Innovative Control Systems to Drive Strategic Renewal*, Harvard Business School Press, Boston, MA.

36. Srinidhi, B. N. 1988. Mathematical formulation of the task segregation problem in internal control system design. *Decision Sciences*, Vol. 19, No. 1 (March), 1–16.

37. Stengel, I., K.P. Fischer, U. Bleimann, J. Stynes. 2005. Mitigating the mobile agent malicious host problem by using communication patterns. *Information Management & Computer Security*, Vol. 13, No. 3, 203–211.

38. Studer, R., V. R. Benjamins, D. Fensel. 1998. Knowledge Engineering: Principles and methods. *Data & Knowledge Engineering* 25, 161–197.

39. Teeter, R., and G. Brennan. 2008. *Aiding the Audit: Using the IT Audit as a Springboard for Continuous Controls Monitoring*, Unpublished working paper, Rutgers Business School.

40. Topaloglu, U. and C. Bayrak. 2008. Secure mobile agent execution in virtual environment. *Autonomous Agents and Multi-Agent Systems*, Vol. 16, No. 1 (Feb.), 1–12.

41. Umble, E.J., R.R. Haft, M.M. Umble. 2003. Enterprise resource planning: Implementation procedures and critical success factors. *European Journal of Operational Research*, Vol. 146, No. 2 (April), 241–257.

42. Vasarhelyi, M.A. 1980. A taxonomization of internal controls and errors for audit research. *Proceedings of the Touche Ross University of Kansas Symposium on Auditing Problems.*

43. Vasarhelyi, M.A. 1984. Automation and changes in the audit process. *Auditing: A Journal of Practice & Theory* **4** (1) (Fall), 100–106.

44. Vasarhelyi, M.A., M.G. Alles, A. Kogan. 2004. Principles of Analytic Monitoring for Continuous Assurance. *Journal of Emerging Technologies in Accounting*, Vol. 1, No. 1, 1–21.

45. Vasarhelyi, M.A. and B. N. Srinidhi. 1989. Adaptation and use of reliability concepts in internal control evaluation. *Advances in Accounting, Supplement 1*, 141–158.

46. Wastell, D. G., P. White, P. Kawalek. 1994. A methodology for business process redesign: Experiences and issues. *Journal of Strategic Information Systems*, Vol. 3, No.1, 23–40.