Principles and Problems of Audit Automation as a Precursor to Continuous Auditing

Michael G. Alles
Alexander Kogan
Miklos A. Vasarhelyi

Rutgers Business School
Department of Accounting, Business Ethics & Information Systems
One Washington Park, Room 928
Newark, NJ 07102-3122

Version: September 11, 2009

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, 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 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 likely to succeed 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, reengineering.

* We thank participants at the 16th and 17th World Continuous Auditing and Reporting Conferences in Newark and Thessaloniki, Greece; the 2009 AAA Annual Meeting in New York, and the 2009 UWCISA Conference in Toronto for helpful comments. The comments of three anonymous referees are also gratefully acknowledged. Additional feedback is appreciated and may be addressed to kogan@rbsmail.rutgers.edu.
1. Introduction

“The first rule of any technology used in a business is that automation applied to an efficient operation will magnify the efficiency. The second is that automation applied to an inefficient operation will magnify the inefficiency.” Bill Gates

Continuous Auditing (CA), in which assurance is provided closer in time to the underlying transaction than in traditional auditing, is only feasible in a setting in which most of the relevant information process is automated, from transaction recording by the firm to the analysis of data and controls by internal and external auditors. The automation of the audit portion of that information process is necessary but not sufficient for the provision of continuous assurance, since the latter encompasses not just doing auditing faster, but also more comprehensively and over a broader set of dimensions (Alles et al, 2002, 2008b). But there is no question that audit automation is the foundation upon which CA is built, and hence, understanding the principles for implementing it systematically and successfully is a priority in CA research and practice. In particular, while by necessity much of the first steps in audit automation consists of automating existing manual audit procedures, the history of automation teaches that transformational change also requires reengineering to a greater or lesser extent: either to rethink those manual procedures altogether, or at least to the extent that it makes it feasible to automate as many of those manual procedures as possible. Otherwise, as Bill Gates warned, automation for its own sake risks simply magnifying all the drawbacks of existing audit practices without enhancing the level of assurance provided.

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 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 as the work we undertook with Siemens on its second generation CA implementation. While the details of that particular case study are 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.

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 that 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.

However, the past experience with automation of other areas of the firm, from manufacturing to government services, provides a warning that implementing automation and achieving the promised return on investment can be problematic. It is that sorry track record that gave rise to the argument for business process reengineering in Hammer’s (1990) article, “Don’t Automate, Obliterate”, by which he did not mean, obviously, that firm’s shouldn’t automate, but rather, that maximizing the benefits of automation requires rethinking what is being automated as opposed to blindly automating procedures originally designed to be undertaken manually.

As we argue in this paper, not only must audit automation be undertaken systematically, it also has to incorporate reengineering in the 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 that Hammer advocated, but this intermediate approach is one that is more manageable from a change management perspective (and more likely to be politically acceptable within the firm) and improves the odds of a successful CA implementation.

Based on our experience, at the beginning of the transition to CA it is going to be more a case of “automate now, obliterate later”, which makes it all the more essential to develop a better understanding of what audit automation encompasses and how it can be brought
about. Reengineering in this context is more an extender of the scope of automation rather than a means of creating a totally new audit environment. That will have to await the development of second generation CA systems that will move beyond simple automation towards a true, continuous or on-demand audit and control platform for the firm and its auditors.

We begin by considering the drivers and objectives of audit automation in Section 2. Section 3 then discusses the formalization and reengineering of audit processes to enable automation. Section 4 introduces a way of enlarging the range of automatable audit procedures by using the technique of baselining. Section 5 discusses the technological infrastructure of audit automation, covering system architecture and software choices and the critical issue of security. Section 6 provides an overview of a variety of challenges and opportunities in audit automation, including change management in its implementation, scalability, and alarm management. Section 7 offers concluding comments.

2. Drivers and Objectives of Audit Automation

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.1 Recent empirical studies (Banker et al, 2002) have shown that information technology indeed leads to significant productivity gains in public accounting practice.2

---

1 The potential impact of audit automation on the changes in the audit process was originally researched a quarter century ago by Vasarhelyi (1984).
2 But significantly, that paper examined the application of technology to the work practices of auditors (the use of laptops rather than desktop computers and moving to electronic work papers) and not to automation of the assurance function itself.
The importance of audit automation and the utilization of IT in modern audits have 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 (at least prior to the economic crisis) received much attention in the professional press, the budgetary pressure on internal audit departments post-SOX was probably even more severe. Thus, it was not surprising that it was internal auditors that took the lead in deploying such audit automation technologies as continuous auditing (Alles et al 2008b).

Furthermore, the progressive sophistication of information technology underlying modern business processes 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. Thus, they do not have to rely on sampling and can process entire populations of transactions instead,
which is another argument for the increased effectiveness of automated audit (Alles et all, 2008a). Additionally, the results of automated audit procedures are available in (close to) real time, thus increasing the timeliness of audit results and facilitating continuous auditing. However, it again has to be emphasized that while audit automation is necessary for CA, true continuous auditing is an outcome of a broader transformation of the scope and domain of assurance, with audit automation as only one, albeit critical, means towards that end.

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 automated analytical procedures (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 was primarily aimed at bringing about CCM, the general principles and problems of audit automation generalized from that experience apply to the implementation of all aspects of CA.

3. Formalizing and Reengineering Audit Procedures

The starting point in most audit implementation projects that we have come across is an existing set of audit programs, usually expressed in a written form as “audit action sheets” or an audit manual. Audit automation in this context means achieving the same outcomes that those manual procedures did (i.e., an acceptable level of assurance) using IT systems that
automate those procedures to the greatest possible extent. In contrast to this approach that pivots off the existing audit program, an alternative is to define the desired assurance outcome and then formulate all new processes that will give rise to it. But for reasons that we discuss elsewhere in this paper, implementing such a fundamental reengineering approach towards audit automation is simply infeasible for most firms. Implementing the more incremental form of audit automation is a two step procedure: putting in place the IT infrastructure that enables automation, and the conversion of the existing audit procedures to make them automatable. It is the challenges in undertaking the latter that we examine in this section.

Since most existing audit programs were originally designed by human auditors for execution by other human auditors who are presumed to largely share their own knowledge and judgment, audit procedures in these programs are not completely formal and leave open significant room for interpretation. This is problematic for the audit automation process since even highly qualified human auditors will often disagree about the precise interpretation of a particular procedure. The resulting lack of consistency is one of the key barriers towards audit automation. Hence, as Alles et al (2006, pg. 156) indicate, before audit procedures can be automated, they must first be formalized, meaning that consensus has to be arrived at on the desired outcome of the procedure before figuring out whether and how that outcome can be achieved in an automated fashion: “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.”

One objection against the formalization of existing audit procedures is the desire to retain the flexibility to interpret it differently according to varying circumstances that the auditor might encounter. Such flexibility obviously necessitates the intervention of a human auditor with the judgment to make a case by case assessment of the procedures, something that is antithetical to the entire object of audit automation. The alternative 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 flexibility that may or may not be taken advantage of appropriately by an individual auditor.
For example, we have come across situations in which auditors sometimes 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 formalizing audit procedures is to anticipate those circumstances where flexibility is truly needed and to make provisions for the appropriate application of human judgment, rather than allowing such judgments to be made idiosyncratically and inconsistently.

Indeed, a key rationale for audit automation is that it is better to use technology to achieve consistency in the application of audit procedures rather than hope that all auditors are experienced, knowledgeable and diligent at all times. Of course, some audit processes can and should only be undertaken by human auditors, but the efficiency and efficacy of the audit as a whole depends on separating those procedures from ones that computers can do more frequently, consistently and with less error.

Having said that, the difficulty in formalizing an audit program should not be underestimated. It can be very laborious and costly since a formal procedure that a computer can undertake has to be defined very specifically and in great detail, and has to describe the precise actions to be performed in various circumstances. 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), which makes it difficult for them, when converting the manual audit procedures, to “think” as a computer would. The fact that most conversions are undertaken by auditors rather than software designers only exacerbates this problem.

To ameliorate the problem of formalization, 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 Fougem et al, 2008). However, we have not come across the use of such technical methods in practice, with most conversion projects being somewhat more akin to unstructured group brainstorming or the simple assignment of tasks
to individual auditors with no formal guidelines for how they should carry out the conversion, other than by simple eyeballing of the manual procedures.

Another complication of trying to convert existing audit programs designed to be processed by humans is that simple procedures amenable to automation and complex judgment based procedures are often intermixed in the same audit manual. Hence, even before formalization and automation can take place, a systematic process is required to separate out automatable audit procedures from those which have to remain restricted to the human auditors applying judgment or specific knowledge—a prototypical example of which is the conversation the lead auditor would have with the senior management of the auditee to assess the “tone at the top”.

Unlike in that example, however, it is not often easy to determine, without further investigation, which category a particular audit procedure falls into. First, whether a procedure is automatable or not may be dependent on the capabilities of the firm’s underlying IT infrastructure. Indeed, when one compares the two rounds of audit automation at Siemens (Alles et al 2006; Teeter and Brennan, 2008), a larger percentage of audit procedures became automatable once the move was made from a home grown audit system to the industrial strength Approva software. Second, while some audit procedures are obviously automatable, and others clearly not, many audit procedures tend to lie in a grey area where the ability to automate requires more detailed analysis to see if by some reworking it is possible to automate them, even when it is not immediately apparent that they are.

For example, an auditor may be instructed to “Retrieve and examine the list of users who have the “administrator” access privileges to a particular system, and determine whether this level of privileges is appropriate for everybody on the list.”3 While this by itself may be sufficient to guide human auditors, who can draw upon their experience and training to determine what is meant by “appropriate”, on the face of it, this audit procedure appears too subjective to be obviously automatable. But that requirement for human judgment can be replaced by a machine implementable rule if the auditors can work with the auditee to specify which category of workers should have administrator privileges—and if the firm’s databases associate

---

employee names with those categories, and the audit automation software could make all the necessary comparisons across databases automatically.

It is this reworking that gives the conversion project an aspect akin to a reengineering of the audit program. However, this step aimed at facilitating audit automation is to be differentiated from a “true” reengineering of the Hammer kind, which would not take the manual audit procedures as the starting point of the audit automation implementation. That “clean sheet” reengineering would not be aimed at converting the existing audit procedures, but would instead ask the question “how should we do this audit if we started from scratch with the intention of automating it”. However, as discussed in the introduction, we have yet to come across any firm that has decided upon such a comprehensive “obliterate, don’t automate” approach towards CA implementation.4

Once a preliminary categorization is made about whether an audit procedure is automatable or subjective, a more detailed procedure is used to determine the exact solution which will bring about automation of the former, which can range from replacement with another [automated] compensating control, to recourse to IT professionals to redesign the software to facilitate that procedure’s automation. In other words, the degree of automation lies on a continuum as opposed to being a yes/no proposition, and correspondingly, the required reengineering ranges in scope and complexity. Obviously, in all these cases, though, the automation should be done systematically (as opposed to ad-hoc) 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.

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.

---

4 In the example above, a clean sheet approach may begin by asking “why do some people have administrator privileges in the first place”, as opposed to taking that as given and simply categorizing workers as administrators and non-administrators.
While formalization is a prerequisite of automation, formalizing an audit program also 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.

As discussed above, an indication of the potential for audit automation is provided by the experience of Siemens IT Internal Audit. Siemens’ methodology for auditing its SAP systems 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, and using a home grown audit automation tool. But Teeter and Brennan (2008), in their study of the CA initiative based on the commercial Approva system concluded that about 68% of the actions could be automated to some extent. Moreover, 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 resulting 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).

In other words, there is the possibility that the more frequent commission through automation of some previously manual audit procedures can reduce the need to rely on other audit procedures that cannot be automated to the same extent. Thus, automation enables the “frequency-scope” tradeoff: the execution of some audit procedures with high frequency can compensate for not executing certain hard-to-automate audit procedures at all. 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 important argument in favor of increasing the proportion of automated procedures in a reengineered audit program is that
these automated procedures can be performed much more frequently than the eliminated manual methods they substitute for.

While we have focused in this section on the broad issues that arise when formalizing audit procedures as a prelude to automating them, we feel that it is still too soon to develop specific guidelines and mechanisms for how formalization and reengineering should be carried out. That would be presumptuous and premature in the face of rapid developments in CA technology, increasing implementations in practice and ongoing academic research. Indeed, such guidelines now would be more of a constraint to the evolution of best practices than an aid to their implementation. After all, there are still no universally accepted standards for implementing much more established technologies, such as ERP systems, or business process reengineering itself. While it is clear that formalization and reengineering will be common features of any audit automation project, now is the time for experimentation to see how to best carry out those steps and to leverage the increasing power of the available technology. As the two Siemens projects indicated, any specific guidelines put forward after the first pilot initiative would have been of limited value by the time of the second, thanks to rapid advancements not just in the audit technology, but also because of the evolution of the audit actions sheets themselves. These have always been routinely revised to reflect changes in the SAP systems, but now—precisely because of the success of the first automation study—these revisions explicitly take into account the possibility that they will be implemented by automation rather than by a human auditor.

4 Automating Audits through Baseline Monitoring

If the evaluation of results of certain important audit procedures is either impossible or cost-prohibitive to formalize, it may be possible to automate subsequent evaluation of results of these procedures using the technique of “baselining”. As applied to enterprise systems, a baseline is defined as a set of system configuration and business process settings at a given, reference point of time. 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. A similar approach provides an alternative, potentially less costly and more easily implementable approach towards audit automation, in particular, as a first step towards continuous control.
monitoring. The benefit of automated auditing through baselining is that it necessitates less formalization and reengineering of existing audit procedures than the more comprehensive approach described elsewhere in this paper, but the drawback is that it also covers less of the audit domain and it is only effective in certain, restricted settings.

Baselining as an approach to audit automation consists in utilizing human judgment for evaluating the results of a certain audit procedure (e.g., certain enterprise system configuration settings) when this procedure is executed for the first time. Thus verified results are then captured and stored in the CA system as the benchmark for subsequent automatic comparison with the results of the audit procedure when executed in the future. Baselining is facilitated through the use of the “snapshot” feature of audit automation software (such as Approva), which does precisely what the name suggests, by capturing in a database all the settings of the SAP tables that the program monitors. This, by itself, is already a very significant degree of automation, since with thousands of tables with numerous setting in each table, capturing this information manually would be a very costly and time consuming task.

Comparing one snapshot with another allows for a rapid and automated identification of those control settings that are different between them. In an audit automation setting the snapshot that is used as the reference baseline can be either the state of the ERP system at an earlier time period, such as right after the last full audit of the organization, or the baseline could reflect the “should be” settings drawn from the audit action sheets. In the former case, baselining would identify the changes that have taken place since the earlier audit, thus directing the auditor to only those areas that need their attention. In the latter case, baselining by itself serves as an audit of the ERP system settings.

Using the snapshot feature potentially allows for the automation of audit procedures that would be difficult to formalize explicitly, by allowing for a simple comparison with a baseline. This approach can be used where the actual activities that the auditee is performing may be difficult to monitor or audit automatically, but where the effects of their actions are likely to be reflected in changes to the tables in the ERP system.

Of course, even this more limited approach towards automation requires an extensive manual effort upfront to verify that all the values in the baseline are appropriate. This verification relies on human judgment typically supported by manual techniques such as
interviewing, investigations and observations, though the effort can be reduced considerably if the existing audit manuals already specify what the control settings should be. The advantage of baselining is that 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 the auditor’s manual verification of every such change is not justified by the risk-cost tradeoff. Generally, the more static the parameters are, the more suitable they are for monitoring through a baselining approach.

Ongoing business operations will inevitably lead to changes resulting in alarms that will have to be 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 details of when the alarm is triggered 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.

Such alarms will 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, thus enabling a built-in trigger for a change management process in audit automation. 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 exacerbated 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 the 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.

We now turn to a discussion of the technological infrastructure of audit automation, and challenges and opportunities in its implementation.

5. Technological Infrastructure of Audit Automation

5.1 System Architecture

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 approach is usually the only option in the case of highly heterogeneous loosely coupled legacy enterprise system landscapes.

Note that the timeliness of data obtained from a data warehouse depends entirely on the frequency with which data enters that warehouse from the firm’s legacy systems. Even if analysis of that data is itself timely, any alarms generated will be referring to events at least as old as the lag in the updating of the warehouse. Indeed, care must be taken to coordinate assurance with data replenishment, for it is possible that the capability of the CA system to
conduct analyses—especially when that capability is automated as opposed to being on-demand—will outstrip the rate at which data can be transferred to the warehouse.

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. There are three reasons for executing audit procedures (whether in the form of EAMs or mobile agents) on the common enterprise platform:

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 sandbox 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 to solve. While numerous solutions have been put forward for tackling this issue (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 resolution has been universally accepted as being able to prevent the host from interfering with code’s execution.

The difficulty 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 may well outweigh 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.

5.2 Software for Audit Automation

While it is certainly possible to design, develop and implement a custom-made automated auditing system in house, the resources 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.

One way of understanding the capabilities of the current software offerings for audit automation is to categorize them using the breakdown of continuous assurance into its components of Continuous Control Monitoring (CCM) and Continuous Data Assurance (CDA). While the vendors are attempting to integrate into 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 incorporate 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.

5.3 Securing Continuous Auditing

Based on the analysis above of the system architecture for audit automation, 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 this 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 “two-pair-of-eyes” controls and the perpetuation 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 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.

6. Implementation Challenges and Opportunities in Audit Automation

6.1 Audit Automation Change Management

The audit profession is inherently conservative—and it has to be—given that its entire value added comes from the auditor’s credible claims of objectivity and reliability. As a consequence, even more so than with other business processes, changing auditing processes requires overcoming a considerable degree 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 surmount. 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 concerns 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 about how such procedures are carried out in practice and what constitutes the acceptable outcome of those procedures, all of which translates into differences in how to convert 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 identify differences in interpretation by utilizing duplicate audit
automation teams to carry out the conversion, and then comparing the resulting automated audit programs. Resolving the inevitable disagreements between the duplicate teams would obviously improve the final automated program, for the same “two-pairs-of-eyes” reason that underlies all auditing. While this approach is often utilized in academic research (for example, 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, other measures have to be utilized to assure the quality of the automated audit program 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, akin to peer review in academia. Another 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, post-automation 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.

6.2 Scalability of Audit Automation

Automating a manual audit program requires a significant startup expense. Since automation of highly specific audit procedures for different enterprise units can incur prohibitive costs, 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. Automation will be economically feasible if it is scalable, and it will be scalable across the enterprise only if the work undertaken on audit automation of one business unit can be leveraged into others without incurring the same high fixed cost.

There are a number of strategies for making audit automation scalable. The most immediate one is the generic parameterization of automated audit procedures. If there is significant homogeneity of enterprise business processes across units—for example, if they have the
same ERP systems, as at Siemens—it is more 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. Having said that, the change management issues involved should not be underestimated because ultimately it is not how technically similar the IT infrastructures across business units are, but how similar managers across those units perceive that their business processes and technology are. And often people perceive what they do as more unique and specialized than it probably is in reality.

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 structuring of automated audit procedures 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.

6.3 Alarm Management in Automated Audit Systems

While it is the ultimate objective of any business enterprise is 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 also have an automated closed loop process for capturing information about the corrective actions and assuring that these actions resolve the underlying problem. Otherwise the alarms will reoccur, thus both driving up costs and driving down the attention that will be paid to them, creating a “boy crying wolf” problem.
While resolving exceptional events and anomalous situations, and identifying control failures of the enterprise system, the auditing system should also 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.

7. Concluding Comments

The practice of audit automation will be strongly influenced by the ongoing software development and maturing of the field of GRC. AMR Research projected that spending on governance, risk and compliance applications and services would top $32.1 billion in 2008, up 7.4% from 2007. 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 may 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 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


