Auditing Applications, Part 2

Tommie W. Singleton, Ph.D., CISA, CGEIT, CITP, CPA, is an associate professor of information systems (IS) at Columbus State University (Columbus, Georgia, USA). Prior to obtaining his doctorate in accountancy from the University of Mississippi (USA) in 1995, Singleton was president of a small, value-added dealer of accounting using microcomputers. Singleton is also a scholar-in-residence for IT audit and forensic accounting at Carr Riggs & Ingram, a large regional public accounting firm in the southeastern US. In 1999, the Alabama Society of CPAs awarded Singleton the 1998–1999 Innovative User of Technology Award. His articles on fraud, IT/IS, IT auditing and IT governance have appeared in numerous publications.

This is the second part of a two-part article on a process-oriented framework for auditing applications. Part 1 (volume 3, 2012) detailed the first three steps: planning, determining objectives and mapping. The remaining steps are described here. The full framework includes the following steps:

- Plan the audit.
- Determine audit objectives.
- Map systems and data flows.
- Identify key controls.
- Understand application's functionality.
- Perform applicable tests.
- Avoid/consider complications.
- Include financial assertions.
- Consider beneficial tools.
- Complete the report.

IDENTIFY KEY CONTROLS

When evaluating the relevant controls, the IT auditor will want to distinguish between customized controls and those contained in commercial off-the-shelf software (COTS). For custom-built controls, inquiry is a good place to begin the evaluation. One of the key questions is to ask management the specific nature of controls expertise being injected into the application development process. That is, who or what group is providing the expertise that makes sure adequate controls are embedded in new applications? How is that goal achieved? And, finally, the IT auditor should make sure those controls have been properly documented and tested.

For COTS, the IT auditor would probably start with a walk-through to determine what controls are actually in the application and how they function. A walk-through would involve following transactions or processes step by step, keystroke by keystroke, with the data-entry person explaining what they are doing and why. Such a process should enable the IT auditor to gain a general understanding of the applications' controls, the adequacy of controls and the nature of them (i.e., effectiveness). This walk-through is especially necessary the first time an application is used by the entity.

Also for COTS, the IT auditor should establish a baseline of controls—tests to understand reliability and effectiveness. These would include configurations for applications, such as SAP and Oracle.

For COTS, the IT auditor needs to determine the responsibility of vendors involved. That goal is why **figure 1**, which is part of the mapping step and detailed in part 1 of this article, has information about the vendor and the nature of maintenance of the application. When a problem occurs with the application, management needs to have assurance of exactly who to rely upon to solve the problem. Obviously, vendor management practices apply.

Figure 1—Mapping Example Using Spreadsheet, Part I

IT	Description	O/S	DBMS	DB Server	Data Location
ABC App	Middleware designed to ...	N.A.	N.A.	XYZ	Birmingham
DEF App	CRM, target ...	Z/OS	DB2	Z mainframe	Nashville

Figure 1— Mapping Example Using Spreadsheet, Part II

Developed	Maintained	Owner	Access Admin	Change Control	Notes
In-house	In-house	Sue Z.Q.	Active directory ...	Controls include ...	Yada ...
Vendor	Vendor, SOC1/2 available	John D.	Security admin ...	Vendor ...	Yada ...

The types of controls can be assessed by using the typical systems model: input, process and output. Input controls include:

- Access security
- Logical segregation of duties (SoD)
- Data validation
- Data integrity
- Coding
- Input error correction
- Batch controls (where applicable)

Typical process controls include:

- The level of automation (e.g., fully automated, IT-dependent, fully manual)
- Job scheduler dependencies (for job processing)
- Job scheduler monitoring
- Auto calculations
- Auto reconciliations
- Auto notifications

Typical output controls include:

- Reconciliations
- Reviews
- Approvals
- Error detection/error reports or lists
- Control over physical reports (ancillary control)

UNDERSTAND APPLICATION'S FUNCTIONALITY

Normally, auditing functionality is a chief audit goal. The procedures involve verifying the operational functionality, which should be described in the information requirements in the application development (AppDev) process. Besides reviewing the authorization document for the application, the IT auditor should review the end-user acceptance report—if one exists. If one does not exist, that says something about the adequacy of control procedures for AppDev: They are lacking a best practice.

Some typical objectives are related to the purpose of the application. When testing the application, consideration is given to the various scenarios needed to properly test the application. If the purpose of the application leads to a dichotomous outcome, a test of one might suffice (yes or no, approved or not approved, etc.). But, if the application is an update to payroll processing, for example, there are a large number of scenarios to consider to test all of the various combinations of factors that go into calculating payroll taxes. The same is likely to be true of testing security and access controls.

Some special considerations include at least a couple of things that the typical end user and business manager tend to overlook in the information-requirements-gathering stage: security and proper scope of data captured. The proper level of security is obviously a critical success factor in AppDev and, thus, needs to be evaluated. Typically, users and managers do not fully grasp the scope of data that need to be captured at the point of events and transactions. This fact is especially important if the entity has any plans to ever employ, for example, business intelligence (BI) or business analytics. A richness of data becomes necessary to "slice and dice" data with data mining tools to gain the maximum benefit of the data in employing BI.

> "The problem can usually be traced back to an improper testing phase."

Operational controls might be in scope, depending on the consideration of purpose. The same is true for financial reporting controls.

Using the system model is likely to make analysis and testing of the application's functionality easier and more complete.

PERFORM APPLICABLE TESTS

When an application fails to perform correctly, when there are errors created, when processes embedded in the application fail to work properly, the problem can usually be traced back to an improper testing phase. Testing the application is more than just performing a single test.

The best practice for testing involves multiple levels of testing. First, the application is tested stand-alone. That is usually done by a senior programmer or analyst who is chiefly responsible for the AppDev project. Then, the application goes through some quality control in the IT department. That is, it is independently tested by some expert in the IT department.

Next, the application is tested by actual users. Often, these end users are involved in a cyclical manner as the application is being developed. But, at a minimum, one or more end users should test the application once it is fully developed in order to determine its functionality, completeness, accuracy and efficiency. After completion, it is customary to have those end

users sign an end-user acceptance report, documenting the results of the test.

Then, the application is tested in conjunction with other applications in the same module, cycle, or class of transactions. That often requires a more robust environment than earlier testing of the application as stand-alone. A staging area has become one of the best ways to perform this test, where a simulator is created of the entity's infrastructure, applications, systems and databases. But, that is not the end either. The application should be tested in the context of the enterprise system, with all of the data transfers and interfacing that goes on in actual IT operations. That process in particular needs a staging area.

AVOID/CONSIDER COMPLICATIONS

There are a number of complications that are inherently risky and, thus, need consideration during the application audit. First, proprietary (custom-built) applications have a high inherent risk. This fact affects the objectives, planning, controls and risks steps.

If a data warehouse (DW) is involved, there is a relatively high inherent risk. Almost universally, when a DW is initially implemented, data being imported into the DW have a high risk due to, for example, inconsistencies in data (same field with different names), missing data and bad data (i.e., errors). Thus, when data are extracted from the transaction processing systems (TPS), care should be taken in mapping the data and using the ETL (extract, transform and load) process to identify and correct the previously mentioned data anomalies.

For the ongoing DW, data owners could, for example, change field names and add fields, and if change controls are not effective, the data cannot pass through the next ETL process successfully. Thus, change management controls for DW are highly important. The same is true for other similar integration functions.

Some distinction should be made between two types of risk with DWs. First, there is process integrity. This integrity is about whether the processing is successful. Does the application do what it should do regarding its processing function? Second, there is data integrity or data quality, which involves the reliability and integrity of the data being processed, transferred and recorded. Were the data entered valid? Are the source data valid, accurate and complete? Was the data transfer from source to target completed effectively, with no errors?

INCLUDE FINANCIAL ASSERTIONS

When financial reporting is in scope, the application needs to address the primary assertions of the account balance, class of transactions or disclosure. Does the application include the appropriate controls related to the primary assertions of the end result account balance or class of transactions? The IT auditor, if applicable, should test the application against the appropriate assertion(s). For instance, if the assertion is accuracy, testing might include things such as:

- Data entry validation controls
- Automatic calculations
- Automatic reconciliations

Existence assertions might be tested for data entry validation controls. Completeness assertions might be tested for job/batch processing controls or reconciliations.

CONSIDER BENEFICIAL TOOLS

Some useful tools for testing applications are computer-assisted audit techniques (CAATs) and ETL. CAATs are helpful in conducting procedures, such as data mining, that examine results in data from posting by the application to determine if the application's controls are working, if the application is working properly and if the application produced any errors. CAATs are also useful in analyzing data for objectives such as data integrity.

ETL is useful in detecting flawed data that can be traced back to the application that produced it and, thus, provide the opportunity to correct the flaw in the application.

Tests of Controls

Some possible tests of controls include:

- Reconciliation
- Recalculation
- Duplication
- Gaps

An example of reconciliation might be verifying the customer ID in the transaction file against the customer ID in the master file. That is, do the customers in the transaction file actually exist in the authorized customer list? Another example is recalculating where the IT auditor might extend the inventory database to see if the total inventory costs match the control total in the general ledger (i.e., the account balance). Duplicates and gaps are useful in detecting errors in data processing.

CAATs

CAATs could be used to reperform automatic calculations or automatic reconciliations.

Data Mining

Data mining could be used to support the audit objectives. In particular, it is useful in conducting IT-related substantive procedures, such as testing approvals or classification errors related to proper codes.

Purchase Order Thresholds

Any time an application involves a threshold where initial/additional approval is needed, CAATs are useful in determining if that control is operating effectively. For instance, if the application is either purchase orders or disbursements, and if purchases and payments are one-to-one (i.e., disbursements are paid by invoice and not statements), a simple test of extracting all disbursements over the threshold against the data file containing the approval (e.g., purchase order file) would expose any exceptions to the control/threshold. This also has the added benefit of fraud detection if someone is frustrating the threshold deliberately to perpetrate a fraud.

Inventory Anomalies

If the app is recording receipt of inventory, CAATs could be used to show whether the application allows zero or negative quantities to be recorded. Obviously that constitutes an error (anomaly) and, thus, the application would be seen as containing a control deficiency and in need of either a change in the application or a compensating control. There are other applications that could make use of this test.

Second, if the application is a file maintenance program, the system would (hopefully) minimize situations in which an employee could make undocumented changes to the inventory data that lead to discrepancies and data errors. Controls are needed to prevent this anomaly. For example, use of logical SoD could limit employees who can make file maintenance changes. Also, the application/system could track changes by recording data before the change and after the change. Without such tracking, employees could falsify changes and create errors or fraud in the data. Data mining could spot differences in account balances by taking the beginning balance, adding up all transactions and verifying the sum against the ending balance. A similar situation exists for any file maintenance application.

> "The successful audit of applications is dependent on a reliable approach."

COMPLETE THE REPORT

Obviously all audits end with some kind of report. Those reports are generally proprietary in format. But, they tend to include the audit objectives, tests conducted, results of tests (usually) and recommendations.

CONCLUSION

The successful audit of applications is dependent on a reliable approach. This two-part article demonstrates a reliable approach and some tools that should be helpful in conducting the audit, especially mapping and CAATs.

ADDITIONAL RESOURCES

Bitterli, Peter R., *et al*; "Guide to Audit of IT Applications," ISACA Switzerland Chapter, 2010

ERP Seminars, "Auditing Application Controls," 2008, *www.auditnet.org/docs/Auditing_Application_Controls.pdf*

SANS Institute, "The Application Audit Process," InfoSec Reading Room, *www.sans.org/reading_room/whitepapers/auditing/application-audit-process-guide-information-security-professionals_1534*