LOG4AUDIT
THE APPLICATION OF LOGGING IN AUDITING AND MANAGEMENT
by
Tatiana Gershberg

A dissertation submitted to the
Graduate School-Newark
Rutgers, The State University of New Jersey
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
Graduate Program in Management
Written under the direction of
Professor Michael Alles
and approved by

______________________
Professor Michael Alles

______________________
Professor Helen Brown-Liburd

______________________
Professor Bharat Sarath

______________________
Professor Gerard Brennan

______________________
Professor Natalia Reisel

Newark, New Jersey
October, 2016
ABSTRACT OF THE DISSERTATION

Log4Audit

The Application of Logging in Auditing and Management

by Tatiana Gershberg

Dissertation Chairman: Professor Michael Alles

The Log4Audit framework, I developed in my dissertation creates a centralized processing engine that captures necessary events from an application log, analyzes them and provides real time monitoring. Based on design research methodology, I find the motivation in the scarcity of studies that utilize large volumes of information contained in system and application logs. The purpose of my work is to provide detailed view of Log4Audit framework I developed and illustrate its capabilities in anomaly detection and management’s assertions guidance.

My dissertation follows the design research methodology. The framework is the first attempt to alleviate the problems of alienated applications employed by an organization to perform different business functions. I provide the theoretical basis to an optimal structure of the Log4Audit framework and application logs that fit a wide range of objectives of a business process. Although there is a wide range of
capabilities of the framework, I focus on its theoretical foundation, implemented tool stack and the illustration of Log4Audit capabilities to provide additional data for audit evidence and assistance in management decision making process. In my observational case studies I determine the robustness of Log4Audit framework. These anomaly detection and management’s assertions case studies illustrate the powerful capabilities of the Log4Audit framework.
ACKNOWLEDGEMENTS

I would like to thank Professor Michael Alles for chairing this committee with utmost dedication, mentorship, support and encouragement. I would like to express my deepest gratitude for the guidance and invaluable contribution of my dissertation committee, Professor Helen Brown-Liburd, Professor Bharat Sarath, Professor Gerard Brennan, and Professor Natalia Reisel. Finally, I would like to thank my family for their endless love and support.
# Table of Content

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>iv</td>
</tr>
<tr>
<td><strong>Chapter 1: Introduction</strong></td>
<td>8</td>
</tr>
<tr>
<td>Design Science Research</td>
<td>12</td>
</tr>
<tr>
<td>Research Statement</td>
<td>14</td>
</tr>
<tr>
<td>Methodology</td>
<td>15</td>
</tr>
<tr>
<td>Contributions</td>
<td>16</td>
</tr>
<tr>
<td><strong>Chapter 2: Research Area and Methodology</strong></td>
<td>19</td>
</tr>
<tr>
<td>Literature Review</td>
<td>19</td>
</tr>
<tr>
<td>Continuous Auditing</td>
<td>19</td>
</tr>
<tr>
<td>Continuous Monitoring</td>
<td>22</td>
</tr>
<tr>
<td>Continuous Auditing and Monitoring</td>
<td>23</td>
</tr>
<tr>
<td>Research Methodology</td>
<td>26</td>
</tr>
<tr>
<td>Design Science Research</td>
<td>26</td>
</tr>
<tr>
<td>Data Mining</td>
<td>30</td>
</tr>
<tr>
<td>Anomaly Detection</td>
<td>33</td>
</tr>
<tr>
<td>Event Log and Process Mining</td>
<td>37</td>
</tr>
<tr>
<td>Log Analysis</td>
<td>40</td>
</tr>
<tr>
<td>Technology Life Cycles</td>
<td>42</td>
</tr>
<tr>
<td><strong>Chapter 3: Tool Stack</strong></td>
<td>47</td>
</tr>
<tr>
<td>Tool Stack</td>
<td>48</td>
</tr>
<tr>
<td>Application Log</td>
<td>48</td>
</tr>
<tr>
<td>Log4j</td>
<td>50</td>
</tr>
<tr>
<td>Disco</td>
<td>54</td>
</tr>
<tr>
<td>Logstash</td>
<td>55</td>
</tr>
<tr>
<td>Elasticsearch</td>
<td>55</td>
</tr>
<tr>
<td>Kibana</td>
<td>56</td>
</tr>
<tr>
<td>Conclusions</td>
<td>56</td>
</tr>
<tr>
<td><strong>Chapter 4: Log4Audit Model as an additional source for Audit Processes and a Management Decision Aid Tool in Automated Business Environment</strong></td>
<td>58</td>
</tr>
<tr>
<td>Architecture of Log4Audit</td>
<td>59</td>
</tr>
<tr>
<td>Design and Toolkit</td>
<td>62</td>
</tr>
<tr>
<td>Data</td>
<td>65</td>
</tr>
<tr>
<td>Performance: stress and loading</td>
<td>66</td>
</tr>
<tr>
<td>Capabilities of Log4Audit</td>
<td>66</td>
</tr>
<tr>
<td><strong>Chapter 5: The Development and Implementation of Log4Audit</strong></td>
<td>69</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

As technical knowledge grows, IT is applied to new application areas that were not previously believed to be amenable to IT support.

  *Signaling the pathway to regeneration.*

The necessity of Log4Audit model (named after a successful and widely used logging utility log4j\(^1\)) appeared as I was investigating the processes of reconciliation of transactional data currently discussed in academic literature. As I further researched into the common enterprise applications that operate transactional data streams I found a necessity to build a unified module that collects the data from different application and serves as a repository of data, which serves as a collector of records.

My dissertation is an attempt to provide an implemented solution to multiple isolated databases within one organization. However, to simply develop a framework that collects data is not an objective of this dissertation. The meaningfulness of the proposed framework is in its usefulness of the data collected to daily business operations. Thus, the purpose of the framework I developed, Log4Audit, is to offer

---

\(^1\) Almost every large application includes its own logging or tracing application program interface. A European research and development project SEMPER decided to write its own tracing API in early 1996. The API eventually became log4j, a popular logging package for Java. The package is distributed under the Apache Software License, an open source license certified by the open source initiative. 
Source: [www.Apache.org](http://www.Apache.org), the Apache Software Foundation that provides support to the Apache Community of open-source software solutions for public good.
the data in its most useful form: easy to explore and query and with minimal latency. My successful implementation proved the architectural stability of the Log4Audit framework and offered real-time data stream to be collected for analysis. The results I gathered underlined the robustness of the Log4Audit framework.

This dissertation is built on the discussions and conclusions I derived from the review of information technologies and audit best practices, benefitting from both domains and providing a broader understanding to the proposed framework Log4Audit and the outcomes of the observations I collected. The interdisciplinary critique and collaboration allowed me to envision the benefits of scientific exchange of practices and theoretical postulates flourishing the discussions and reinforcing the conclusions of my research work. I support Log4Audit framework with the accounting information systems research, information systems research and software development process and its best practices.

In my research I faced challenges of embedding a relatively young research domain of accounting information systems (Solomon and Trotman, 2003). With the exponential increase of incorporation of technology in accounting and auditing processes, the constantly evolving state of computerized business operations generates and stores large volume of data augmenting the research domain of accounting information systems. Due to the increased need to incorporate the use of technology in accounting and auditing, research that explores how methods or tools
can be used to access and utilize the large volume of data in accounting information systems is important and timely.

Vasarhelyi and Halper proposed in 1991 Continuous Process Audit Methodology (CPAM), where the “data flowing through the system are monitored and analyzed continuously (e.g., daily) using a set of auditor defined rules”. Teeter at al (2014) suggests that the audit will subsequently utilize intensive logging of company internal and external activities. The logging framework I present will “help to clarify variations from original processes” and provide support to audit practices (Teeter at al, 2014).

However, the literature abundantly discusses and offers many tools to assist a company with continuous auditing and monitoring (Dull et al, 2006; Alles at al., 2004; Groomer et al., 1989; Kogan et al., 1999; Vasarhelyi & Halper, 1991). Alles, Kogan and Vasarhelyi in their 2005 paper offer a “black box (BB) log file” to “guard the guardians”. Such application of logging in continuous auditing is based on a read-only access to secure records of actions of internal auditors to provide the “audit trail of an audit”. However, the literature addressing the capabilities of a system and application logs is very limited and fails to address the abundance of solutions already available for implementation and advancement of secure logging with minimal latency for producing necessary analytics. Modern enterprise information systems provide a capability to store events in a log file or system of files, often referred to as
“transaction log”, “audit trail“, or “history”. I close this gap with Log4Audit framework.

In practice, current state of auditing and monitoring is a top-down sample based periodic approach. PCAOB (Public Company Accounting Oversight Board) defines top-down approach as one that begins “with the auditor’s understanding of the overall risks to internal control over financial reporting” (PCAOB 2007, para. 21). Although PCAOB identifies in Auditing Standard No. 5 Appendix B Benchmarking of Automated controls, the reality of auditing profession is the sampling method used to test controls and account balances, further analyzing where key controls are identified by a top-down approach from selecting significant accounts, identifying processes, systems and after all controls, to determine the necessary key controls. I realized that this approach does not address some of the important issues of many companies: fraud risk, compliance, assurance monitoring and, furthermore, assumes that the processes in place are followed.

My work is motivated by the scarcity of studies that utilize the large volumes of information contained in application logs. I define the purpose of this research as to illustrate several ways to utilize logs for audit and management purposes, although the framework capabilities are useful for any type of assurance monitoring, fraud detection and prevention, compliance, operational audit, strategic audit, etc.

My first goal is to provide a foundation of the optimal structure of the application logs, supported by the investigation into the logs dataset to determine the
necessary attributes and components of logs to record the entries that allow further analysis.

My second goal is to explore an analytical model that determines the level of severity for anomalies defined by a rule-based expert system proposed, presenting an analysis of application logs in order to disseminate the anomalies.

Lastly, to illustrate the proposed model “in action” I propose Instant Notification Algorithm that assists stakeholders in anomaly detection and management assertions. The Instant Notification Algorithm includes a closed loop escalated alerting to ensure an action taken for each alert, either through an automated algorithm, manual resolution or an escalated mitigation.

**Design Science Research**

The Design Science concept was first introduced in 1969 by Herbert Simon in *The Sciences of the Artificial*, where he explained “the principles of modeling complex systems, particularly the human information-processing system that we call the mind”. Simon (1969) wrote: “schools of architecture, business, education, law, and medicine, are all centrally concerned with the process of design”.

However, I find that the design itself has been a focus of study mainly in technology driven disciplines, such as computer science and engineering, unlike
accounting information systems that only recently accepted design into behavioral research.

Although information systems research objective is trifold, to investigate the interaction among people, organizations and technologies, accounting information systems research omits the analysis of the interaction between people and technology, and importantly the design of technology itself (Hevner, March, Park, & Ram, 2004).

“The integration of the processing logic found in computers with the massive stores of databases and the connectivity of communication networks” to include “IT infrastructure, innovations with technology, and especially the Internet” led to the evolution of the research area that enabled design science research (DSR) (Agarwal and Lucas, 2005). In contrast, I find that other researchers have referred to design science research as “exploring by building” (Vaishnavi and Kuechler, 2007) and dispute the ability of design science research to answer questions arguing there is scarce or non-existent theoretical support. I assert that design science research is the most appropriate methodology for design, implementation and evolution of application of logging in auditing.

A vast quantity of studies focus on proper design of IT artifacts, resulting in effective and efficient outcome:

- Brohman et al. (2009): implementation of a network-based customer service systems framework,
- Wu (2009): development of a flexible form-based knowledge for problem solving and exploiting activities,

- Reinecke and Bernstein (2013): offered a prototype web application to support cultural adaptivity.

Therefore, I focus on illustrating and providing details and analysis to the implementation of incorporation of additional information, such as logging, in continuous auditing and monitoring. After my extensive review of the literature, it appears to me that the application of design science research in accounting information systems is still in its infancy with most publications in information science journals (Arazy, Kumar, and Shapira (2010), Kuechler, and Vaishnavi (2008), Geerts (2011), Gregor, and Jones (2007)). I note that this scarcity of extant design science research in accounting information systems underlines the necessity to extend the application and testing of design science research methodologies into the accounting information systems domain.

**Research Statement**

To meet new demands of a fast developing computerization of business operations, I suggest that the research in accounting information systems should embrace the abilities of design science research. Given that design science research literature provides enough evidence to demonstrate the ability to solve problems and produce effective IT artifacts (March & Storey (2008), Gregor & Hevner (2011)), I
propose a framework with the support of the design science research methodology to illustrate the incorporation of additional knowledge derived from the logging activity in a business environment to assist in management and auditing procedures. Thus, I pose the research questions with the goal of providing and testing a logging framework for auditing and management purposes and offering observational cases studies:

1. Can Log4Audit, an application of logging in auditing and management, be undertaken successfully?

2. How the proposed model of application of logging in auditing and management is designed, developed and implemented?

3. What are the abilities of Log4Audit framework?

The goals and research questions I pursue through three main phases: (1) the theoretical and methodological background to implementing logging analysis for auditing and management purposes, (2) Log4Audit design and built, and (3) observational case studies analysis.

**Methodology**

With an extant research in accounting information systems and software development practices, I offer a logging framework design and its implementation, as well as two observational case studies. My work is based on Design Science Research methodology. The resulting data I collected are then used to illustrate the Instant
Notification Algorithm I develop specifically for Log4Audit to assess audit and management capabilities and to provide the analysis of the exploration raised in the research question above.1

**Contributions**

My work contributes to the accounting information systems discipline in many ways. First, I provide a unique alternative to a common procedure for data retrieval and analysis for management and audit purposes. The Log4Audit framework I propose benefits management and internal auditors by aggregating large amounts of data for real-time analysis and monitoring.

Secondly, in my research I find that today continuous auditing and monitoring procedures are still largely restricted to a sample database extraction. The PCAOB, in Staff Inspection Brief of October 2015 states: “firms are moving in the direction of developing and using software audit tools to analyze full populations of accounting and financial reporting transactions to select transactions for testing rather than selecting a random sample of transactions for testing”. The real-time analysis, a hallmark of the Log4Audit framework I propose, performs with the least latency and yields valuable information to stakeholders. My further investigation of framework’s capabilities provides foundation to a preventive2 audit and monitoring. To offer a demonstration, I build Log4Audit to collect the data and further develop a methodology for its analysis, Instant Notification Algorithm.

---

2 Preventive audit assumes measures taken to block activities prior to their execution.
Thirdly, with Log4Audit in place and under the future vision of continuous monitoring, the framework I developed internal auditors and managers will have real-time access to the entire database, looking at the underlying financial statement and obtaining data from real-time monitoring tools. In summary, I find the Log4Audit framework offering real-time monitoring and ex-ante alerts with a broader spectrum of information, unlike a common approach of utilizing logs to reconstruct past events to assist in auditing and management tasks.

Unlike prior research, the log files fed into my Log4Audit framework are extracted from a production environment with framework implemented prior to the collection of data, thus fully implementing the framework and embedding it into a production environment ahead of any data collection and analysis. My contribution is also in expanding the horizons of log utility for audit and management purposes — the research literature lacks analysis of log files, and rather focuses on a process mining of a structured extraction archival data from a database, or an event log. Event logs mining does not allow real-time analysis and loses abundance of data stored within an application, thus lacking the preventive component of a robust real-time monitoring. Event log follows a process instance only and is comprised of ex-post data. My Log4Audit framework permits real-time monitoring and ex-ante alerts. Thus, I propose framework that expands the capabilities of continuous monitoring currently implemented in practice.
Further, I take a step to implement aggregation and utilization of the information internally, already collected within an organization. Although the Event Log contains the metadata and permits tracing the relationship of an event to other events and related parties, Log4Audit framework expands the volume of information available for analysis and monitoring, thus increasing the precision of the output of analysis and timeliness of the information processed.

I propose the Log4Audit model that utilizes the information that contains sensitive information, which is kept private and protected (such as personal customer information, banking and credit card information). Thus, I find no studies that have demonstrated the importance and the impact of inclusion of such information into continuous monitoring and analysis. Additionally, I find that a smaller set of events permits a more detailed analysis of detected pathways of each process. No prior research have studied a relatively small set of data that allows to look at all paths and see if it is more contributive to look at outliers (common) or at the full set of events.

Lastly, my dissertation is the first attempt in implementing a well-established technique in computer science to log all the necessary information to discover the evidence of a process into accounting and audit. The two case studies I provide offer illustrations and details to other researchers conducting work.
Chapter 2: Research Area and Methodology

Literature Review

Although the focus of my work is the development, implementation and illustration of usefulness of Log4Audit, the foundation of my research consists of two areas: continuous auditing and continuous monitoring. Here, I offer an implemented solution to automate such tasks as anomaly detection and suggestion of management assertions.

Continuous Auditing

With the active embedding of technology in business processes, the impact of technological advances is observed in the audit process as well. In the 2001 paper titled *The impact of information technology on the audit process: an assessment of the state of the art and implications for the future*, the authors summarize the effect of envisioned commonplace use of paperless audit processes in audit planning, testing and documentation (Bierstaker et. al, 2001). In automated business environment, computers generate internal control templates to help identify and improve the system weaknesses.

In contrast, in traditional non-automated audit procedure of controls testing is performed by an auditor on a scheduled periodical basis. The significant drawback of this approach is in the delayed analysis of the business activities. I find that to adjust
the business process in a timely manner in response to the testing results, the output of audit procedures should reflect the current state of the business.

Even with the proper technology in place, auditing procedures still utilize a snapshot of data, rather than a real-time data stream. In the current fast-paced financial markets, backward-looking audit is rarely of any value to a business performance or regulatory compliance due to testing of outdated numbers for accuracy (i.e., analysis of a snapshot of data) (Coderre, 2015). Shifting the gears into the automated mode, significantly decreases the cost of compliance. In 2011, Ponemon Institute Research study *The True Cost of Compliance, A Benchmark Study of Multinational Organizations* stated that the cost of compliance averages to US $3.5 million per a multinational organization in order to comply with Sarbanes-Oxley Act\(^3\), Payment Card Industry Data Security Standard\(^4\), state privacy and data protection laws. This cost is incurred due to a lack of automation and intensive manual processes, involving external and internal sources.

The response to the increasing need in automation of auditing activities is the expanding interest in automation and technology. Imbedding intensified due to fraud scandals tarnishing names of such corporations as WorldCom, Enron, Adelphia, Tyco, Lucent, and Xerox. However, any level of automation of the audit procedures needs to

---

\(^3\) Sarbanes-Oxley Act - United States federal law enacted July 30, 2002 to protect investors by improving the accuracy and reliability of corporate disclosures made pursuant to the securities laws, and for other purposes.

\(^4\) Payment Card Industry Data Security Standard (PCI DSS) mandated to create an increased controls to cardholder data in order to reduce credit card fraud. Source: PCI Security Standards Council
address the internal control testing and testing of transactions. A financial statement audit is “an examination of an entity’s financial statement, which have been prepared by the entity’s management for external users, and of the evidence supporting the information contained in those financial statements”, defined by Porter, Simon and Hatherly (2014), and any automation of the process would first and foremost focus on satisfying the most basic requirements. Thus, continuous auditing is “a comprehensive electronic audit process that enables auditors to provide some degree of assurance on continuous information simultaneously with, or shortly after, the disclosure of the information” (Rezaee et al., 2002)

The Canadian Institute of Chartered Accountants and American Institute of Certified Public Accountants define continuous auditing as “a methodology that enables independent auditors to provided written assurance on subject matter using a series of auditors’ reports issue simultaneously with, or a shot time period of time after, the occurrence of events underlying the subject matter”. Although I find this definition, as well as in published works of continuous auditing leading researchers, indicating that the continuous auditing methodology enables external auditors, Information Systems Audit and Control Association Standards Board suggests that such reports are beneficial and should be used by both, internal and external auditors.

Continuous auditing undeniably increases the assurance of the reports within a reasonable amount of time, extensively expediting the delivery time with the help of implemented technological advances. Disregarding the purpose of the output, any
continuous audit implementation, for external or internal audit purposes, is vitally
dependent on a well-oiled processes in place with a properly embedded technology.
The “continuity” of continuous audit is in my opinion essentially dependent on two
aspects: the real-time availability of the necessary stream of data and necessary near
real-time tools to analyze the stream of data and produce the required output.

**Continuous Monitoring**

Continuous Monitoring (CM) “is a process to ensure that policies and
processes are operating efficiently and to assess adequacy and effectiveness of
controls” (CICA/AICPA, 1999). Continuous monitoring commonly refers to the
processes that permits management to ensure that business policies and procedures
are followed and effectively performed. Continuous monitoring is performed and
analyzed by the management layer of a business infrastructure. Thus, it is the
responsibility of the management to provide and assess the requirements and
adequacy of controls. Important to note, these controls either mimic or significantly
overlap the controls of continuous auditing utilized by internal auditors (KPMG,
2009).

Similar to continuous auditing process implementation, reports, alert
notifications and other tools assisting a business management tasks are designed to
assess different indicators that ensure the quality of business processes and their
adherence to the rules and regulations in place, and thus promoting the overall quality
of control systems. Opposite to continuous auditing I find, the responsibility for proper implementation and adequate delivery of these controls lays on the management layer of a business hierarchy.

**Continuous Auditing and Monitoring**

While I find that there is no requirement or necessity to have continuous auditing and continuous monitoring implemented, the advantages are well known to the finance and audit executives. As discussed above and summarized in the table below, continuous auditing and monitoring have different audience and goals.

The uniting aspect of continuous monitoring and auditing is achieving the business goals for the organization itself through:

1. Rapid adaptation to risk and regulatory climate,

2. Decreased duplication of efforts and controls and increased cost-effectiveness and coordination of internal audit and management requirements,

3. Transparency in independence of an auditor and management and their responsibility for assurance and performance of controls.

David Coderre summarized the effectiveness of continuous auditing and monitoring implementation in his report, “the audit activity’s approach to, and amount of, continuous auditing depends on the extent to which management has implemented continuous monitoring” (Coderre, 2015). Comprehensive monitoring of
internal controls significantly reduces the audit effort, while the lack of management effort monitoring of internal controls increases the effort and resource volume involved in audit effort. Thus, I conclude that not only continuous monitoring provides an insight into business operations, but also diminishes the audit effort.

<table>
<thead>
<tr>
<th>CA is an automated, ongoing process that enables internal audit to:</th>
<th>CM is an automated, ongoing process that enables management to:</th>
</tr>
</thead>
<tbody>
<tr>
<td>collect from processes, transactions, and accounts data that supports internal and external auditing activities,</td>
<td>assess the effectiveness of controls and detect associated risk issues,</td>
</tr>
<tr>
<td>achieve more timely, less costly compliance with policies, procedures, and regulations,</td>
<td>improve business processes and activities while adhering to ethical and compliance standards,</td>
</tr>
<tr>
<td>shift from cyclical or episodic reviews with limited focus to continuous, broader, more proactive reviews,</td>
<td>execute more timely quantitative and qualitative risk-related decisions,</td>
</tr>
<tr>
<td>evolve from traditional, static annual audit plan to a more dynamic plan based on CA results</td>
<td>increase the cost-effectiveness of controls and monitoring through IT solutions</td>
</tr>
<tr>
<td>reduce audit costs while increasing effectiveness through IT solutions</td>
<td></td>
</tr>
</tbody>
</table>

Source: A report by Deloitte, "Continuous monitoring and continuous auditing: From idea to implementation” (2010)

Table 1: Comparison and contrast of continuous monitoring and continuous auditing.

Importantly, the continuous auditing and monitoring relies on the continuity of data stream. The data should be highly available for analysis purposes either in real time or near real time (Flowerday et al., 2006). Any interruptions of data stream cause
delays and, thus, lower quality of reporting. Therefore, a continuous auditing and monitoring solution should provide the following features:

1. adaptability to various data formats,

2. capability to retrieve data from different data sources,

3. simultaneous analysis and testing of system controls and transactional data,

4. verification of data integrity

Although the implementation of technology in auditing and monitoring processes is vital to their quality, the processes remain the same, providing adequate assurance of management activities and the quality of business operations. Auditors examine management activities, verify controls quality, provide recommendations to manage risks, while management assess the effectiveness of controls and determine associated risks, while improving the business processes. With automation of accounting system, real-time assurance is a necessary provision. However, to provide a comprehensive continuous auditing and monitoring system, integrated technologies need to be solidified into one model in order to provide auditors and management with reliable and immediate necessary results.
Research Methodology

The basis of my dissertation lays in the design science research. Proper information technology research intertwined with critical understanding of necessary changes in continuous auditing and monitoring necessitates a deep focus on understanding the problem, building and artifact, evaluating its usefulness and communicating results and findings.

Design Science Research

To properly understand design science research, I find it necessary to study first the two paradigms of acquiring knowledge: behavioral science and design science (March and Smith 1995). Behavioral science seeks to develop and prove principles and laws, which “explain or predict organizational and human phenomena surrounding the analysis, design, implementation, management, and use of information systems” (Hevner et al., 2004). Design science provides a mechanism builds and evaluates an artifact through multiple iterations before a final artifact is achieved (March et al., 1995; Markus et al., 2002). Benbasat and Zmud (2003) describe an IT artifact as “the application of IT to enable or support some task(s) embedded within a structure(s) that itself is embedded within a context(s).” As design science research is applied to the development of artifacts, the motivation of the researcher becomes what ought to be, rather than what is (Simon, 1996). As March and Smith (1995) suggest, the output of design science research is not necessarily a
fully functioning information system, but rather a set of primary component outputs, such as constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices), and instantiations (implemented and prototype systems).

Information systems researchers have paid most attention to the complexity of relationships among people and organizations by utilizing behavioral science toolbox and neglecting the artifact itself (Orlikowski et al., 2001). In recent years, I note that the research refocused on issues of design process and the importance of a robust artifact (Gregor et al., 2011; Hevner et al., 2011; March et al., 2008).

Hevner et al. (2004) introduced “concise conceptual framework and clear guidelines for understanding, executing, and evaluating” design science research. However, preceding Hevner, a paper published in 1990 by Takeda, Veerkamp, Tomiyama and Yoshikawa was the first introduction of a design cycle of conducting a design science research. The authors distinguish an object, where a designer thinks about the object itself, and an action level of the design cycle, where a designer thinks how to proceed with implementation. Thus, indicating two main steps of a design cycle: identification of a probable solution and design of a solution.

Currently, the five-step process is distinguished, beginning with Awareness of Problem, then Suggestion, Development, Evaluation, and Conclusion. Kuechler, Vaishnavi, and Kuechler Sr. (2007) stated that the awareness of a problem can arise from new developments in industry or within a specific reference discipline.
Awareness of a problem arises when a problem is picked and the object is compared with specifications of a problem. Thus, enumeration of problems is complete and decision on a problem to be solved is presented. This step results in key concepts suggested to solve a problem — this indicates a Suggestion step. During a Development step, the goal is to produce such candidates for a problem from key concepts that are best fitted to produce a solution. An Evaluation step is an implication to a Development step — evaluation of all candidates takes place to assess their structural computation, cost assessment and possible behaviors. The last step is Conclusion, where decision is made to adopt a candidate with any necessary modifications.

However, I find one of the most cited design science work, Design Science in Information Systems Research by a group of researchers, Hevner, March, Parka and Ram, in 2004. To my opinion, the work’s one of the most detailed developments is the Design-Science Research Guidelines. The purpose of the guidelines is to “assist researchers, reviewers, editors, and readers to understand the requirements for effective design-science research” (Hevner et al., 2004) Importantly, the cornerstone of any knowledge and understanding of a design problem is gaining it from the process of building an artifact. In continuation of the evolution of design research methodology, Peffers, Tuunanen, Rothenberger and Chatterjee developed a Design Science Research Methodology in their 2007 paper with the intention to incorporate principles, practices, and procedures of design science research and provide a model
of the outcomes through a consensus-building approach, blending in well-established approaches of various disciplines.

Contrasting design science with natural science permits me seeing another angle of understanding of the essence of design science research. Herbert Simon in *The Science of the Artificial* book makes a clear distinction between "natural science" and "science of the artificial" (also known as design science):

- *A natural science* is a body of knowledge about some class of things -- objects or phenomenon -- in the world (nature or society) that describes and explains how they behave and interact with each other.

- *A science of the artificial*, on the other hand, is a body of knowledge about artificial (man made) objects and phenomena designed to meet certain desired goals.

Thus, Natural Science Research objective is to explain phenomena while Design Science Research objective is to develop ways to achieve research goals. The difference also can be seen in the main output of natural science — theory; while for design science the output are constructs, models, methods, and implementations. Design science creates artifacts, brings about phenomena (which in their turn become targets of natural science research) and provides tests of the claims of natural science research.
Data Mining

Throughout my research, I used various data mining techniques to understand the business process and the data the business application generates better. Data mining (other terminology commonly used is data discovery or knowledge discovery) is the process of analyzing data, discovering useful knowledge and summarizing it into new useful information. Data mining software produces results in return to an open-ended queries submitted by a user. In order to retrieve proper information, it is important to pose the right questions in the form of. This can be done in a number of ways, and it is important to ask the right questions in order to gain information that is interesting and usable (Hay et al., 2008).

Data mining involves large linking separate transactions and analytical systems (e.g. machine learning). The process of data mining I followed consists of five main steps:

1. extraction, necessary transformation and upload of data;
2. storage and management of the data;
3. management of the access to the data for analysis purposes;
4. configuration and analysis of data with applicable tools;
5. visualization and other forms of presentation of data in necessary forms.
Prior to execution of any data mining algorithm, my first step was defining and understanding the goal. It is the most definitive step: adequate volume of data, required structure of data, type and parameters of query and access. Data mining algorithms require proper setup of the following aspects: type of input data, necessary normalization and transformation of data, efficiency and scalability of a given algorithm to the input data and required output, as well as desired adjustments along the data mining process should also be possible (Chen et al., 1996). Preparing data for the data mining is a challenging task as well and should never include any reduction in the data as it may cause prejudice and reduce the quality of new discoveries (Chuvakin, 2013). Thus, my research suggested that depending on the data, isolated clusters of events, single events, uncommon combinations, or suspicious counts of common events could be defined as rare events (Hay et al., 2008). Once the results are obtained by applying a proper method of data mining, a visualization tools are utilized to present the results in an easy to interpret and utilize format (Hay et al., 2008). Following these steps significantly increases undersignanding of the business process and allowed me to proceed to further refining of the Log4Audit framework. For example, the structure of the types of users altered the chain of permitted events of any user. I discovered that not all users are able to make a purchase and different types of users have different permissions within an application.

---

5 The non-disclosure agreement (NDA) with the company providing access to the application, resources and live data stream prohibits any release of the details of any aspects of business process and underlying technologies and supporting resources, except those directly involved in a credit card processing.
Furthermore, an important aspect of data mining is whether the data is obtained in real time or on historical basis. In *Stream Data Mining and Anomaly Detection* Esmaeili and Almadan (2011) present their work on anomaly detection where a data mining methodology is applied to two environments: real time data stream and a snapshot of historical data. Although the work is aimed at improving network security of the environment, one of the take-aways of the paper is the change in the behavior of the model depending on the data type. Given the calamities of continued use of historical data to alleviate current issues and achieve better auditing and monitoring results, I abandon any consideration of archaic data use in Log4Audit.

Practitioners divide the goal of data mining procedure into two types: descriptive and predictive mining. Descriptive mining is focused on describing the given data set within its domain; predictive — to forecast trends and probabilities within predictions derived from the data. While during the development of the design of the Log4Audit framework I relied on descriptive mining, predictive mining was dominant in the refining stages. Regardless of the approach, there are four main types of relationships sought for: clusters, classes, associations and sequential patterns. Clusters are data points grouped together according to a logical relationship or requirement. Classes, or frequent item sets, are stored data is analyzed to determine the points in predetermined groups. Sequential patterns describe what objects occur in a specific sequence (Hay et al., 2008). Finally, associations are different patterns and parameters related to each other or a group of data points. Finding association and sequential patterns was the main focus in the observational cases of Log4Audit.
From the technology standpoint, the main two factors I considered that I find affecting the performance of a data mining application are: size of the database and complexity of a query. More powerful IT system can handle more processing requests and is capable to process and maintain larger database repositories. The scalability of Log4Audit framework permits access to multiple sources of data stream in the form of activity messages to be logged.

Anomaly Detection

Identification of anomalous sequence of events, one of the observation case studies I offer as an illustration to Log4Audit capabilities, is an important task pertaining both continuous auditing and monitoring that has long required a better approach to automation. With Log4Audit, I offer an engine that is capable of precise analysis of full data stream and complete detection of anomalous sequence of events. Anomaly detection (also referred to as outlier detection or deviation detection) is a procedure to find cases (data points) deviating from the rest of the data (Hodge and Austin, 2004). Recent process of anomaly detection required an expert user to be an expert in determining anomalies by hand (Hill and Minsker, 2010). With the automation of business processes, new methods of anomaly detection automation evolve demising the costly involvement in this procedure (Hodge and Austin, 2004). The spectrum of anomaly detection has also widened — now expanding to an intrusion detection system, fraud detection and system security (Bolton and Hand, 2001).
Best practice I analyzed suggest that selecting and determining the fit of the algorithm to the data and its scalability are the starting point (Hodge and Austin, 2004). Currently, I found and taken into account when developing the architecture of Log4Audit that the data stream configuration becomes an important stage of architecture in an algorithm built; and system processing power and timeliness of data exchange are the deterring factors to anomaly occurrence (Gunter et al., 2007).

Hill and Minsker in their *Anomaly detection in streaming environmental sensor data: A data-driven modeling approach* work published in 2010 offer a real-time anomaly detection method where environmental data streams are used to identify data that deviate from historical patterns. This method does not use any abnormal data to make predictions in order to alleviate potential deviations abnormal events could introduce. Authors also note the importance keeping in mind how the alarms raised by the anomaly detection are analyzed and handled. Patcha and Park (2007) state that intrusion detection systems are “predominantly signature-based intrusion detection systems that are designed to detect known attacks by utilizing the signatures of those attacks”. The authors conclude that such systems must utilize near real-time updates for rules and signatures. Although anomaly detection is a subset of intrusion detection systems, where “zero day” attacks are possible to detect, high false alarm rate is still a hurdle to overcome. In live environment alarms, false or not, are guided to be checked and verified (Yamanishi and Maruyama, 2005). I address these findings by implementing a closed loop alerting algorithm.
Another important point I find in the literature is necessity to ensure proper fit of the method chosen for anomaly detection, it is necessary to understand what data is available, including its size and parameters. Although there are many different methods, scientifically developed and commercially available, the major three groups are: supervised, unsupervised and statistical methods.

The goal of supervised anomaly detection is to obtain, or “to learn,” a mapping from an input to an output. Thus, pre-classified data is essential to define expected and anomalous behavior. The drawback of this type of anomaly detection is the absence of anomalous cases in the training dataset leads to their omission in the live environment (Hodge and Austin, 2004). However, the wider the spectrum of cases included in the training data set, the higher the quality of detection is. For example, classification rules present both, abnormal and normal behavior and describe the expected behavior for both types of patterns (Lee et al., 2001).

In contrast, unsupervised anomaly detection only input data is available and analyzed. The result of unsupervised learning is a data point that does not fit the detected regularity of the input. For example, clustering model provides groups of data points with similar attributes. Clustering model allows detection of outliers without determining or obtaining any prior information about outlier. Thus, unsupervised learning permits detection of previously unknown types of faults (Bolton and Hand, 2001).
Artificial neural networks are used for both prediction and classification problems. The architecture commonly consists of input layer, hidden layer(s) and output layer. The usefulness of artificial neural networks application in anomaly detection is ability to recognize unknown patterns and to learn complex behavior. The disadvantage is in the training of the model and potential overtraining it.

Anomaly detection process involves outlier analysis, where outliers are the only focus. Thus, anomaly detection heavily relies on statistical methods and hundreds of outlier tests (Hodge and Austin, 2004). The importance of data preparation cannot be missed here — different statistical methods require proper understanding of the data structure and needed output: mean, variance and other parameters, quantity of outliers and their characteristics, other requirements and expectations. In some cases the distribution of the data may be unknown and thus the fitting is required in order to find the right fit (Knox and Ng, 1998).

Importantly, the urgency of incorporating the newly available data into analysis is still not transformed into the real-time anomaly detection solutions. Although statistical methods are capable of producing robust results for the live stream of data, the fine tuning is challenging, although not impossible. Veasey and Dodson (2014) offer a solution where the p-value can be used to rank the newly appearing data points according to how farther parameters are from the common and thus move to near real-time analysis of the data stream. Although the data stream
available to me was limited to one month of collected data, Log4Audit architecture evolved and tested within a live data stream requirements.

However, the same stream of data might require different approaches depending on a business model and other aspects. For example, although can be different depending on. Another approach I determine that can be done when performing anomaly detection is dividing data into different parts, and analyzing those parts separately. Lee, Kim and Rhee (2001) suggest that dividing log files into groups based on the historical data to determine similarities and differences that affect the analysis of data stream allows the fining tuning to achieve better accuracy of analysis (e.g. time of the day, day of the week). In my dissertation, although not the goal of this work, I took these probable differences in user behavior into account when analyzing the business processes.

**Event Log and Process Mining**

In contrast to Log4Audit framework, currently, an event log is obtained by a stakeholder to analyze and produce necessary reporting. An event log is a record of system generated events which are chronologically ordered, contain information about originator, activity and its processes. The most recent and expanding body of accounting research literature examines the application of process mining of event logs. I find event log application as a rapidly deteriorating approach that will eradicate itself with the current preference to continuous and real-time auditing and monitoring.
Auditing relies on data and its analysis, validating business processes and their execution to meet requirements and regulations not only set by governments, but also managers. Process mining was first researched focusing on software engineering process in 1995 by Cook and Wolf. In 2010 Gehrke and Mueller-Wickop developed an algorithm for mining financial processes and a methodology for extracting the data from ERP systems. The result of the extraction is a systematically organized event log.

An event log is a “sequentially recorded collection of events, each of which refers to an activity (well-defined step) and is related to a particular case (process instance)”. An event log is mostly comprised of at least the following characteristics: a timestamp, an activity, a case, and an originator of the event. A timestamp is necessary for a chronological sequence of a string of events to determine the preceding and following events that constitute the process. The format of a timestamp is unified throughout an event log. A case ID is unique for every process instance, unlike an activity (e.g. sign, submit, pay). Additional information is added to the log in order to develop a proper level of the This information is extracted from a system database and compiled into a structured table. An event log relies on a set of “interrelated documents and items” that represent each financial business process instance. Thus, only accounting data is taken into consideration to compose an event log.
Process mining is a technique developed to assist management in analysis of business processes by applying data-mining algorithms to event logs, an extraction from structured and unstructured data source. The purpose of an event log is to contain all necessary data to show events and all necessary attributes that describe the order, manner and other characteristics of each event, as well as their sequence. In other words, an event log is composed of events, their time stamps, referred to a specific process and an instance (or a case).

Table: an illustration of an event log.

Consider Table as an example of an event log. With only four records, it is easy to interpret the meaning of this table. The columns indicate a time stamp, a case ID, an activity ID and an originator to each event. Starting with the first record, at 1:04 pm an activity #5643 was performed by an employee or an automated service with an ID #8756 working on case ID #123. Each ID is listed in a library and is available for deeper interpretation. Also, from the same log snippet one can see that the same originator worked with the same case ID, and the same activity was performed for two different cases. Depending on the structure of the log, it can be very expressive or very scarce. The most efficient log is the one that contains only necessary information to complete the task. The log containing excessive information possesses a threat to efficiency of an analytical algorithm and its scalability of data stream.
Three different perspectives are distinguished as a starting point for mining: the process perspective, the organizational perspective and the case perspective. Determining the perspective is necessary to properly identify the process analysis techniques.

The process perspective predominantly concentrates on the control-flow, or the ordering of activities, to find a good-fit model to all possible paths. The process perspective allows to determine the sequence (and probable deviations) of activities for a type of a case. The organizational perspective focuses on the originator’s involvement in a process itself. This perspective allows to structure the architecture of an organization by classifying people in terms of roles and responsibilities. The case perspective’s interest is in properties of cases — paths common to a process, or originators working on a case. However, all three analysis perspective are often related and rarely looked at in isolation (Van der Aalst et al., 2007). Although I utilize the process mining technique to identify the patterns of customer behavior, the algorithm of finding a sequence of events for each user and subsequent matching to the known paths is automated within the Instant Notification Algorithm.

**Log Analysis**

Given the tightened interest to the analysis of logs, many commercial products are available to process the information contained in different types of logs. The range of available tools is very large, from specific needs of a technology to wide analysis
of business operations. Sumologic or Logentries, for example, allow thresholds to be set for specific alerts of abnormal behavior or inactivity, while Google Analytics Intelligence Events is another example of log analysis, with web traffic automatically logged with different attributes and providing indicators to significant statistical changes in the usage, location and traffic data. LOGalyze collects all log data in one repository, provides analytics and permits events and alerts by correlating any log data collected (Zuriel Kft, 2016). Splunk specializes in operational intelligence and log management by gathering machine data, processing stand providing visualization and alerts. Splunk is able to automatically extract data with predefined fields, alerts and statistics, and any further functionality or data that does not fit those formats must be manually input (Splunk Inc., 2016).

<table>
<thead>
<tr>
<th>TimeDate</th>
<th>CaseID</th>
<th>ActivityID</th>
<th>Originator</th>
</tr>
</thead>
<tbody>
<tr>
<td>0503120816</td>
<td>123</td>
<td>5643</td>
<td>8756</td>
</tr>
<tr>
<td>0503120818</td>
<td>123</td>
<td>4563</td>
<td>8756</td>
</tr>
<tr>
<td>0503120818</td>
<td>124</td>
<td>2456</td>
<td>231</td>
</tr>
<tr>
<td>0503120819</td>
<td>125</td>
<td>2322</td>
<td>9003</td>
</tr>
</tbody>
</table>

Table 2: Event log fields: TimeDate, CaseID, ActivityID and Originator.

Yamanishi and Maruyama (2005) offer their model considering the dynamic nature of data that requires the model to adjust in real time. They implement series of hidden Markov models with dynamic learning of parameters and checked against threshold values to determine anomalies. It is necessary to utilize several different
techniques in order to adjust to new and widely different aspects of the process (Bellec et al., 2006; Brugger, 2004).

The importance of understanding the current state of log analysis utility is necessary to understand the full value of Log4Audit. The framework I offer does not analyze the log intrinsically. The framework I developed “listens” to different sources for the statements indicating activities I determine necessary to be analyzed, splits and indexes these statements, then stores, for faster and more intricate analysis.

Technology Life Cycles

Understanding the underlying methodologies and techniques is only one way of discovering the necessary knowledge for robust architectural implementation of Log4Audit. Following the proper technology life cycle is fundamental to timely and strategically build a firm foundation for Log4Audit. Linden and Fenn (2003) states: “several technology life cycle models attempt to measure the evolution of a technology. The two most popular are:

1. the maturity curve, which shows the increase in a technology’s performance over time, and

2. the adoption curve, which shows market adoption over time.”

The last is not measured in regards to Log4Audit. The framework is not commercially on the market or available for any other organization to implement.
However, for a forward-looking development of any technology, I find it necessary to take into account all developmental aspects.

Technology adoption is segmented into five stages in order to organize the thought process of how people accept an innovation and is categorized by adopters: innovators, early adopters, early majority, late majority, and laggards. Each stage represents a group of people adopting a new technology. In the case of Log4Audit, I find the technology adoption to be at innovators level. This segmentation of technology adoption was first introduced in 1962 by Everet Rogers in “Diffusion of Innovation” book and is still widely accepted as a benchmark. In the book, Rogers proposes that the four factors influencing the adoption of a new disruptive idea are: the innovation itself, time, social systems and communication channels.

The maturity curve shows the development of a technological changes as it matures, where different stages of maturity are maturity levels: embryonic, emerging, adolescence, early mainstream, mature mainstream, legacy, and obsolescence. Given the Log4Audit has been only implemented in one organization, the maturity of my framework is embryonic.

Technology life cycle is an integral part of a strategic technology plan of an organization. Gartner (2007) defines a strategic technology as a technology “with the potential for significant impact on the enterprise in the next three years. Factors that denote significant impact include a high potential for disruption to IT or the business, the need for a major dollar investment, or the risk of being late to adopt”.
The new invention period, or the embryonic stage, is the first stage of the maturity curve. This stage is signified by slow initial growth, most experimentation and research is done and technology design is worked out. Given the infancy of Log4Audit, I determine it to be in the embryonic stage.

Once the adolescent, or the invention period is complete, the technology improvement, or emerging period begins characterized with rapid and sustained growth. My further research determined at the adolescent stage, technologies are imbedded in application environment and commercialization takes place with increasing number of vendors. Organizational adoption is assisted with analysis of improvements and regressions within an implemented application. It is a common practice to see surveys or interviews with experts and vendors. For a multi-generation technology, it is in behavioral research that an organization finds limitations or strengths in technology developed.

During the next phase, the mature-technology, or early mainstream, the technology becomes vulnerable to obsolescence and substitutions with fewer opportunities to progress in performance and effectively obtain new or better performing technologies. However, an organizations posses invaluable experience with the technology and offers best practices. Here, researchers often find the best case studies for best and worst practices and implementation studies.

As technology progresses, a better understanding of the strengths and weaknesses of technology usage become more apparent to an organization. I am
looking forward to working with the organization to determine further evolution of Log4Audit.

As applications go mainstream and a technology progresses further from early mainstream to mature mainstream where a larger set of organizations adopt technology. Due to availability of a larger scale of adoption, empirical analysis becomes feasible to determine different measurements impacting the adoption of technology. At this stage, research indicates that design science loses its value in as many of the major problems have already been addressed prior to this stage.

Mature mainstream brings about robust and time-tested technology with very limited evolution on technology and customer side. This stage is of a great interest to researchers as issues associated with technology adoption and diffusion can be studied using empirical research. Mature mainstream has no or very little impact from design science applications.

The legacy maturity level does not introduce any new developments as the costs are restricted and migration to newly available technologies is not feasible. An organization at this stage is focused on maintenance of steady revenue. A technology is widespread and the stability of it is achieved to its most significant levels.

As well as legacy, obsolete technologies pose very limited interest from design science and information systems research. On absolute maturity level, technology is rarely used and new technologies have taken a larger market share.
Log4Audit, although still in its early stage of infancy, will progress and develop through all the stages as determined in my research. However, from my analysis of information technology, auditing and management domains I find neither similar concept nor technological advantage Log4Audit offers.
Chapter 3: Tool Stack

In computer science, software developers have been guided to utilize log files in order to trace different events occurring within the computer system, which include and are not limited to errors, critical information, and warnings. At the very minimum, logged information helps to point to lines of code that produces an error. Although there are different log outputs (e.g., a console log, a database log, a file log, a syslog), the standard purpose of logging is to capture necessary events in order to assist daily tasks of development and support of an application. Reading a log file with a naked eye can be a tedious and often an impossible task. However, to easily operate and understand the information at hand, there are many commercial applications written to support the process of knowledge discovery.

The transfer of the purpose of logging in software development to the domain of auditing and management of a company was one of my motivations to create Log4Audit framework.

Importantly, even the most sophisticated tool will not produce meaningful results to a log file specialist if they are not familiar with the business process and technical aspects of logging. To properly log the information necessary, first the requirements are gathered to understand the business needs. Second, a data base model is studied to extract the data and omit unnecessary clutter of the output. For a
live data stream, additional work may be required to ensure an interrupted stream of data.

**Tool Stack**

Tool Stack, or solution stack, is a necessary set of components to support architectural choices and requirements in order to implement and support an application. Guided by design choices, an architect provides recommendations about an operating system, database structure, programming language, servers and other components. Throughout my research of the underlying methodology and concepts of logging and auditing procedures, I developed a list of most advanced and fitting components that perform tasks needed to implement Log4Audit capabilities.

**Application Log**

Logging best practices suggest there are many types of logs: application, security, server, and database with many sub-types depending on the need and structure of the business process.

In my dissertation the application software produces log files. Application software is any program, or a group of programs, designed for the end user. Application software is commonly called end-user software and is built on such concepts as database, word processing, spreadsheets, or web browsing. An application log file, therefore, contains all events pertaining the use of an application software.
Importantly, what is written in an application log is determined by a business process and implemented by a software engineer of an application software. In developing Log4Audit, I tightly collaborated with the development team and business stakeholders of the application. However, the operating system itself and users of the application software (other than software engineer) were not involved in the implementation of the required structure of the log file.

To properly understand the information logged, it was important to me to understand the structure of log files and information gathered.

Technical literature indicates that the simplest, and most widely utilized, type of logging is based on writing the events into a text file. I inspected the credit card processor log file and determined that following the best practices of logging, this log indeed had every event recorded in each line of the file. Every subsequent event is written in the line below the preceding one. Each line represents an event of a payment submitted to the credit card processor. The first line contains all the column headers of the log. Thereafter, the rows represent every submission of a payment. In a comma separated values file, every column value is separated from the next with a comma. However, I realized that knowing and understanding the structure of a log file clarifies the procedures and the business logic of a customer’s attempt to pay for a product or service, which is challenging without applicable tools.

Another type of log file is a text-based log file format where each event is logged within several rows. This type still is represented by a structured format. Such
logs are more complex to analyze and interpret by a specialist without tools, convoluted by a dependency among the events logged in a file. The business application I integrate with mainly works with logs of this structure.

Also, a different type is a binary log, most structured and only readable by a tool. For example, an event could look as a string of four bytes “B7 02 00 00”, where the length and the alpha-numeric combination differs one from another. This log is virtually crash-safe because only complete events are recorded and retrieved. In my work, I do not incorporate these log files.

Many computer data logging packages offer various frameworks used to standardize, ease and streamline the logging process. In my dissertation, the theoretical basis of logging and log file is set by a Java-based logging framework called log4j, where a log file is an output of a log4j framework.

**Log4j**

Log4j is a Java-based\(^6\) logging framework. I determined that the three main components important to scrutinize to configure a log file are loggers, appenders and layouts. These three components permit fine-tuning of a log: enabling and disabling messages of different types and levels.

---

\(^6\) Java is a programming language and computing platform first released by Sun Microsystems in 1995. Java is fast, secure, and reliable. It is designed to enable development of portable, high-performance applications for the widest range of computing platforms possible. 
Source: [www.java.com](http://www.java.com)
The logger component captures events for logging and stores them in accordance with name entities given as logical log file names that follow the hierarchical naming rules with such parameters as ancestors, descendants, parent and child relationships, which are easily determined. Loggers permit an application to log the events regardless of the output destination. Importantly, a logger may be given a verbosity level (when level is not assigned, a log inherits the level according to its hierarchy) such as: TRACE, DEBUG, INFO, WARN, ERROR, FATAL. The granularity of logging changes from the most fine at the TRACE level to the most severe at the FATAL level. TRACE is commonly included in the log file to detect parts of a function (a type of procedure or routine that returns an output to a given input) and is not a common setting. When TRACE level is indicated to be included in the log file, the logging procedure will include also DEBUG, INFO, WARN, ERROR and FATAL levels. While Severity Level is associated with each log message and is an indicator of importance of information contained in the message, Verbosity Level of a log determines the inclusion of all messages with the corresponding Severity Levels.

The next component of a log4j framework is an appender. The main responsibility of appenders is managing the output destination. Such destination, among many, as discussed above, can be a file, a database, a socket, or a console. In short, appender component determines how a log is stored. After I studied the framework of log4j, I discovered the flexibility and defined the task of creating a new appender that would integrate with Log4Audit framework. Thus, I determined the "route" to connect Log4Audit with the business application I worked with.
The final component of the log4j framework is the layout. At this stage, a message is transformed to adhere to a custom layout, ranging from a simple level-message structure to an HTML (Hyper Text Markup Language) or XML (EXtensible Markup Language) highly customizable layouts.

Figure 1: Tool stack of Log4Audit.
Figure 1 describes Log4Audit components responsible for search and analytics of the data. A logger is represented by log4j module, an appender - by logstash instance, indexing the data, and elastic search applies searching capabilities to produce the search result to be visualized in a custom application for data visualization Kibana.

The three components of the log4j framework I configured with utmost regard to the business process and model. Throughout my trials to configure the proper settings, I noted that the log file written with the highest verbosity level can cause a lag in an application performance due to a large body of data pushed to be logged (at DEBUG level, all events with the following severity levels are included: FATAL, ERROR, WARN, INFO, DEBUG). Although there are ways to prevent or lessen such interruptions in service (e.g. queuing logging, asynchronous logging), it is business processes and a business model that determine the best fitting practice of logging. However, the cost of computer memory and computing power has been decreasing (McCallum et al. 2002), (Rashid et al. ) (Hwa Chung et al. 2000) and therefore giving an opportunity to embed a logging practice into many business processes.

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>The OFF has the highest possible rank and is intended to turn off logging.</td>
</tr>
<tr>
<td>FATAL</td>
<td>The FATAL level designates very severe error events that will presumably lead the application to abort.</td>
</tr>
<tr>
<td>ERROR</td>
<td>The ERROR level designates error events that might still allow the application to continue running.</td>
</tr>
<tr>
<td>WARN</td>
<td>The WARN level designates potentially harmful situations.</td>
</tr>
</tbody>
</table>
Table 3: Common levels of logging.

**Disco**

Fluxicon Disco is an intelligent process mining tool that helps to visualize process data in a very short period of time by providing insight about a process from raw data. Disco brings process transparency by automating process discovery, providing detailed statistics and multiple filters. The goal of Disco, short for Discovery, is to offer all necessary information in order to improve processes. This tool helps to visualize the process and gaps it produces.

With other tools on the market, Disco is the open-source, easy to use, and most importantly build with the efficient and precise process mining algorithms and log management.

Important to note, I chose Disco to only utilized for the purpose of process discovery in my dissertation. However, the process of paths identification and their alignment against the common paths was automated to accommodate the live stream of data. The convenience of this tool offers necessary process discovery and does not
impede the real-time data stream characteristic of Log4Audit I implemented in live environment.

**Logstash**

Logstash is a data pipeline that helps you process logs and other event data from a variety of systems (Source: www.elastic.co). I chose Logstash because of its unique approach connecting to a variety of data sources and streaming to a central analytics system. The choice I made allowed Log4Audit to be easily adoptable for other business systems and processes. An organization is likely to have many data sources, contained within different department, vendors, or geographical locations. To make these volumes of information useful to an organization, Logstash allows to stream these different data sources to a single engine powered by Elasticsearch.

**Elasticsearch**

The main purpose of Elasticsearch is to allow search and analyze data in real time. Minimal latency was my priority while still choosing the best available solutions in informational technology. Elasticsearch is a commercial product that has dominated the market with its real-time advanced analytics, scalability, failure resilience, security, and other great features. At this stage of my work on Log4Audit implementation, one capability, full text search, was necessary for proper analysis of the data and scalable architecture of Log4Audit. Full-text search supports “multilingual search, geolocation, contextual did-you-mean suggestions,
autocomplete, and result snippets” (Source: www.elastic.co). I utilize logs within Log4Audit that contain different types of data. These logs are not extraction of financial data, but meta-data of processes that take place creating different statements, entering and modifying data.

**Kibana**

Another important component I added to Log4Audit is a visualization interface. Data is often represented by a stream of events that are hard to visualize. Kibana gives form and shape to large volumes of data, structured and unstructured. I selected Kibana due to its natural of integrating with Elasticsearch. The two products are the leaders in its fields and developed to be seamlessly integrated when needed.

**Conclusions**

While I find the logging procedure and its initial molding to the business needs is an integral part to any SAAS (Software-as-a-Service) business, I suggest the proceeding search and analytics influence the company’s logging infrastructure. As mentioned above, a log file is only a medium of attaining a goal posed in front of a manager or an internal auditor. Ultimately, an investigation of a log file with a help of searching and analytical tools leads me to a discovery of a process as a string of events:

\[
\text{Process} = (\text{event}_1, \text{event}_2, \ldots, \text{event}_{n-1}, \text{event}_n)
\]
Determining the string of events is possible due to the attributes that permit mapping one event to another, building a chain of events that depicts a process. Thus, a Process is comprised of a strictly arranged in a chronological sequence order of events.
Chapter 4: Log4Audit Model as an additional source for Audit Processes and a Management Decision Aid Tool in Automated Business Environment

Enterprise resource planning (ERP) systems are currently the most common enterprise management solution used by companies to gather, store, and manage data from such processes as finance, HR, procurement, manufacturing, service, and sales. (Kurbel, 2013) ERP system optimizes all business critical processes.

The main purpose of integration of an ERP system into any organization is to tightly interweave a single solution of a business unit (such as financial, accounting, manufacturing or human resources) into a common platform that promotes information flow throughout the entire organization. Importantly, ERP systems allow expanding into the systems of suppliers and customers, thus increasing the volume of data collected. The purpose of the centralization of such collection of data leads to significantly accurate and timely data analysis.

However, the current approach to finding a business solution to respond to business needs shows a departure from a common one provider to catalogue cloud-based solutions. Thus, a company is likely to require an integration with multiple sources of activities necessary for auditing and management of a company. Thus, although currently ERPs are dominant on the market, I will further refer to the originator of user activity as business solution software.
Architecture of Log4Audit

Business solution software is implemented generally in large and medium size enterprises to conduct business processes and transactions. Business solution software systems hold and unify the data pertaining to these processes and transactions in a uniquely structured database. Among many benefits of business solution software systems, the efficiency of conducting a business has driven the extensive implementation and collection of data through business solution software (Sayana et al, 2004). The database, the heart of any business solution software system, collects data from many business applications and modules, commonly grouped as follows: manufacturing, marketing and sales, purchasing, accounting and finance, and human resources.

Figure 2: Log4Audit architecture.
In its core, it was important to me to design the Log4Audit model (Figure 2) as a centralized processing engine that captures events, analyzes them and provides real-time monitoring. Considering the evolution of technology (tools and machines), capturing events associated with every piece of data on a granular level is within today’s technological capacity. Reaching conclusions through valid reasoning by auditors can be supplemented with artificial intelligence providing exact set of events that led to an accounting record being examined. This new data produced by Log4Audit can tell a story of what led to this consolidated data and help internal auditors identify potential issues based on facts. Furthermore, I find that certain patterns can be recognized by machines and alert not only internal auditors, but also management layer of the organization (further I develop this idea into Instant Notification Algorithm). I specifically ensured that Log4Audit framework also allows for customization of analytics: Search Engine and Audit Message Indexer, for example, enable tuning of search for keywords in certain proximity, or adjusting verbosity or severity of logging.

Business solution software already utilizes a logging framework, which frequently consists of application logs, system logs and other components that record different type of activities. Log4Audit framework subscribes to listen to events that originate within a business solution software environment. An event data is then submitted to the logger. Logger’s main function is to accept messages, accompanied by a date and time stamp and severity.
However, recording all activities can be very cumbersome, unnecessary and costly. Playing a role of a proxy between a business solution software system and Audit Message Indexing Engine, Logger can also be adjusted to the verbosity level. My architectural decision allows Audit Message Indexer pull the event data from the Message Queue and serialize it by indexing. Indexing was another important solution to large volumes of data. Indexing (or tagging) the data processed within Audit Indexing Engine, assists with structuring the data and producing higher quality analytics.

Further, Audit Database component functions as a repository of indexed structured and unstructured data. My design allows this data repository to be accessed by a search engine in order to produce analytics that satisfy the needs of Business Intelligence tools and Real Time Monitoring to generate meaningful output that I further give necessary access to be investigated by internal audit or management of the organization.

Importantly, when building a robust architecture of Log4Audit, I decided it was necessary to take into account the latest trends in the world of business solutions: a relatively large ecosystem of cloud service providers is becoming a trend and increasingly a choice of many companies over a monoculture. Thus, looking forward, the integration required an event handling capability of Log4Audit framework to incorporate an observer pattern of log4j implementation, widely used in many modern applications to handle logging tasks. Subscriber, Log4Audit, “listens” to notifications
from a source of activities and decides whether and how to log these events. This architectural decision allows minimal adjustments to new solutions of business daily operation automation.

**Design and Toolkit**

In Log4Audit model, I only included application logs. I took this approach to minimize the effects of information contained in other types of logs (e.g. system breadcrumbs in a system log). Additionally, I believe this minimalistic approach permits a precision in the procedure identification to obtain a proper illustration of a relationship between a business process and logging instances.

The business model of the company, which application I embedded Log4Audit with, requires a user to login to the website to make a purchase. I illustrated the process in Figure 3, where a user activity logged is indicated with a mark `log.info`. Through analysis of logs and database structure together with a deep understanding of the business process, I determined a series of logged activities. When a user adds an item to a shopping cart, another logged event is recorded. However, when a customer is ready to pay for the items added to the shopping cart, a Credit Card Processor receives the necessary information entered by a user for processing a payment. Important to note, the payment process in its entirety (from the moment a customer clicked on Pay by Card button to the successfully processed, or failure to process, payment notification) is logged by the only entity involved, a payment processor.
Figure 3: User Registration Activity Diagram. A database structure diagram in relationship to the business process from logging perspective.
The credit card processing does not happen within the company's application. Credit card processing activities is a domain of a PCI DSS compliant third-party provider. Thus, to identify the failed and processed payments, a browser sends a credit card token to the credit card processor’s server. Simultaneously, the payment method information is pushed to a Global Payment Platform and then the Payment Processor server. Once the payment is settled or denied, the card token with the status of the payment is returned to the browser. Once the token is received, the application log records the status of the order.

Figure 4: Credit Card payment processor model

Figure 5: Flow of transaction data of Credit card payment.
Data

I collected the data through the implemented Log4Audit in a software-as-a-service business environment. The organization is a company with a product at an adolescent maturity stage with customers in North America, South America and Australia. The data is accessible for analysis in real-time through Amazon Web Services Kibana application.

For analysis, I first extracted the data with the help of Kibana Application. Kibana request I wrote for fetching data for further analysis is as follows:

message: "User successfully logged" OR message: "Add ItemGroup1 to Shopping Cart...Ok. Account ID:" OR message: "Add ItemGroup2 to Shopping Cart...OK. Account ID:" OR message: "Add ItemGroup3 to Shopping Cart...OK. Account ID:" OR (message: "Order successfully settled. Order No:" AND message: "Account ID:" )

Although for demonstration purposes the imported file into the process mining tool was extracted, the Log4Audit framework successfully runs with full capacity to provide the real time analysis.

The two data sets I found necessary for analysis purposes to satisfy the following illustrations. I determined the first dataset to be composed of application logs within Log4Audit framework. The second dataset I extracted from a credit card processor application logs. The application logs are limited in its time frame: from

---

7 Due to the non-disclosure agreement in place, the Kibana request is anonymized.
April 3rd, 2016 to May 3rd, 2016. However, when I applied process mining tool to the first dataset, I identified records of 18491 events and 6446 cases.

**Performance: stress and loading**

One of the most important requirements of the model is its performance. A modern business entity could employ thousands of devices interconnected in real-time and generating a large amount of events simultaneously. In a dissociated logging architecture, unlike the Log4Audit framework I propose, consolidation and unification of data repository is a highly costly objective. I designed such logging infrastructure in order to build on a stand-alone logging nodes with internally-focused goals. Log files produced in this infrastructure are highly differentiated in its structure and therefore difficult to aggregate. Thus, the Log4Audit model proved itself to be flexible in adapting to different business solutions.

**Capabilities of Log4Audit**

The capabilities of the Log4Audit I designed are unlimited due to its technological advances and architecture. However, I find the focus of my dissertation to be on addressing the issues currently lacking the research and implementation, yet highly important for a manager and an auditor of a company. I define the uniqueness of the model in its centralized data repository and analysis of the gathered data.
This framework is endemic to auditing and management purposes due to the following aspects:

- aggregation and analysis of large volumes of data in its entirety available for investigation of past events,

- real-time analysis and alert system built on rule-based algorithms with prediction and prevention capabilities,

- full search and analytical capabilities with real-time indexing of incoming event messages,

- higher quality of analysis and prediction of event outcome due to correlations of event data,

- ease and flexibility to adapt to a catalogue cloud-based solution providers,

- closed loop escalation alert system enforces an action taken to each alert,

- better accuracy and latency of analysis of real-time data.

Log4Audit is not a set of tools. Log4Audit is a concept with a proper architecture in place. The value of Log4Audit is not in its technological stack, but rather in its framework that promotes real-time capture and analysis of events that lead to meaningful conclusions and suggestions: behavioral analysis of consumers, predictive analytics, market trend analysis, and other. The componentized architecture of Log4Audit allows me to replace any of the pieces with a better solution on the
market. Thus, the framework can evolve and scale depending on the budget and needs of an organization.
Chapter 5: The Development and Implementation of Log4Audit

SDLC Stages

Software Development Life Cycle, or SDLC, is a set of generally developed methodologies used to develop software. Each stage within the selected software development cycle is determined by the set of activities: identifying the scope of the concept and the necessary resources; gathering requirements and performing their analysis; prototyping and modeling the database; developing and refining software; testing code and performing regression analysis; deploying software code to the live environment and testing; maintaining software. Log4Audit is a stand-alone application that I designed and architected through every stage of development. The necessary code was written by the development team.

The timetable and the sequence of software development life cycle phases is different from one to another. Although the same type of the cycle can be very different from project to project, the common types of software development life cycles are: waterfall, v-shaped, evolutionary prototyping, spiral, iterative and agile. Main purpose of any software development life cycle is to create a structured process to design, create and deliver quality software with the help of well-defined various tasks.
The software development methodologies vary across organizations. ISO/IEC 12207 Systems and software engineering - Software life cycle processes specifies the following steps: acquisition, supply, development, operation, and maintenance. These standards provide guidance for all the tasks necessary to develop and maintain software. I utilize SDLC process to help produce a product that is cost-efficient, effective, and of high quality.

Acquisition phase, also called as planning and requirement analysis, starts with the initiation and proposal preparation. On the business side, such activities as contract negotiation take place as well. In this stage many management and planning tasks are completed to ensure the cost-effectiveness and the timetable of the product delivery. The main goal is to acquire as many details of each requirement as necessary and to ensure the scope of the work is clearly determined. Commonly, such tasks as testing and the manner of delivery is also recorded. In the case of Log4Audit, I met with the team to determine the requirements and plan the processes ahead of the start of development.

Importantly, I found that the methodology chosen by the product management team greatly affected each stage because different approaches can be taken in moving from one phase to another. For example, waterfall model requires all software requirement specifications need to be finalized and documented prior to moving to the next phase. In contrast, an iterative life cycle does not require a start of a new phase to
be coordinated with a completion of full specification of requirements. The team I worked with utilizes agile methodology.

Next phase, design, software engineer and architects initiate a high-level design of the product, incorporating the requirements into delivery of each requirement. Here, the role of the software engineer and architects was placed on me with final approval and coordination of efforts from the management team. At this stage, technical details I communicated (and adjusted if necessary) to stakeholders to alleviate potential issues concerning risks, technological stack, team structure, project constraints, time schedule and budget. An architectural design assumes all components are defined for development, third party services are engaged, database connectivity and user interaction is explained.

To implement the requirements and design, the next phase of the software development life cycle begins with software engineers beginning their work with writing code in accordance to requirements and architectural specifics determined in earlier stages. To cover different aspects of implementation, special skills are required to create the necessary data structure in the database, user experience and user interface software engineers create interfaces and graphical user interface to create a “bridge” that permits necessary communication between a user and a back-end of an application. In iterative model, a segment of a process of developing an application in cycles until the requirements are met is called an iteration. At this stage I was
available to brainstorm and answer any questions concerning the architecture and provide clarification of any requirements in place.

The following phase of the software development life cycle is testing. This phase is very important for successful deployment and use of an application by a customer. During this phase, I verified all requirements against an application, I reproduced and carefully documented any defects for further correction. Testing also validates the necessary behaviors of an application. Once a fix became available, I tested the Log4Audit framework again. These steps were repeated until all requirements were met and thoroughly tested.

The next stage is deployment and release. Once the confidence in the quality of a product was sufficient enough for a release, the team deployed the framework on a production environment. Although the development process seemed to be over, support and necessary modifications were still part of work. Post-production issues do not necessarily signal low quality product, but rather novelty of utilization of software development processes on my side.

**Security of the framework**

Ensuring security of the Log4Audit framework, its data repository and analytics model is important to promote its reliability. If an application is not reliable, any derivations or output resulting from that application unlikely to be viewed as trustworthy. Woodroof and Searcy (2001) suggest SYSTRUST (or a CA derivative of
SYSTRUST). ISO 17799, an information security standard published by the International Organization for Standardization could also be used. However, there is a number of best practices and standards that address the security issues.

The lack of accuracy and consistency of data invalidates any robustness and intricate implementation of the most sophisticated frameworks. However, continuous monitoring has addressed the security of the data to provide assurance of data.

Gartner\(^8\) anticipates “27 percent of corporate data traffic to bypass perimeter security and flow directly” to the public cloud. With many information security technology already in place, businesses deal with increasingly advanced attacks. Security best practices always evolve to include new technological advances and anticipated threats. Log4Audit is built to withhold most of the attacks and secure external penetrations of the framework. First, the server log (Log4Audit runs on a server) records the activity of services sending requests from Log4Audit. For added security, commercially available solutions can monitor the activity externally as well. In case of downtime of Log4Audit, logging for security purposes, external monitoring records necessary activity prior, during and after a possible penetration of the framework. Secondly, proactive and continuous scanning of activities reacts to unknown or harmful events. Either type of defense and prevention can be adjusted to

---

\(^8\) Gartner, Inc. is the world’s leading information technology research and advisory company that provides technology-related analysis and insight to high-tech and telecom enterprises, investors, IT leaders and enthusiasts.
the level of action taken: from passive alert notification to an active shut down of the framework.

According to Gelmalto, the world leader in digital security, “the main motivation for cybercriminals in 2014 was identity theft with 54% of the all data breaches being identity theft-based, more than any breach category including access to financial data”. To address the possible threats, Log4Audit does not collect any credit card information and utilizes encrypted in-rest database. All credit card information is processes within the premises of a Payment Card Industry Data Security Standard (PCIDSS) complaint credit card processor, while all the personal information is stored with a cryptographic protocols that provide privacy and data integrity of the highest standard between any two applications.

Incorporating large datasets into traditional methods of audit expands its validation procedures beyond accounting data analysis. Reliance on a recorded transaction becomes outdated as more data floods the data storages without its contribution to the auditing. Researches and practitioners are no longer refining the traditional audit procedures, but rather seek to compliment its quality and integrity with supplemental data analysis derived from external sources and ultimately achieving full benefits of big data use in validation process. Integration with large data necessitates utilization of informal types of information and thus moves auditing further from its traditional dogma.
Log4Audit infrastructure supports vast sizes of unstructured data (and thus, big data) and promotes the support of a standardized recording of a transaction with additional data. Vasarhelyi et al. (2010) determine that when metadata is implemented with transaction data, audit efficiency and efficacy improves. Now many resources are utilized to extract data for further analysis and is commonly auto collected and recorded. For instance, traditional data could be linked to security, news, media programming videos and recordings. Thus, assurance should imply the necessary assimilation of big data into its procedures, forfeiting sampling of the population in exchange for full population analysis.

With the size increase of the data necessary for analysis and required for storage, manual management has been replaced with auto-tiering to accommodate its scalability and instantaneous availability. Assurance process implies high-quality prevention of unauthorized access and should necessitate provision of continuous availability to access the data when necessary. However, an auto-tier storage management system does not provide homogeneous security. Thus, I determine that an audit log recorded earlier and not accessed for an extended period of time could be distributed to a lower-tier with lessened security without any consideration of sensitivity of information it carries. This exposes sensitive datasets to malicious attacks and possible loss of confidentiality of information.

To ensure the security of data, a dashboard is comprised that allows an internal auditor as well as any other user permitted to access the Log4Audit system to approve
a level of security and thus augment the auto-tier management system to improve the security of data.

First, to understand the specifics of the application of proposed model, I discuss a procedure further. A designated user logs into the system to perform an activity. Creating a profile of different level of access that may require a security level assessment prior its active pooling and migration to data storage is an essential step. An authorized and authenticated user is presented with a set of options that allows the system to estimate the level of necessary data security:

i. frequency of accessibility,

ii. time period of accessibility,

iii. response time to input and/or output operations,

iv. assigned security/privacy rating,

v. role access level,

vi. geographical restrictions.

Frequency of accessibility implies the level of access required for the continuous auditing procedures to run smoothly and without delays. However, the data access should not be granted on a need or continuous basis. This notion is imperative to security of data as well as efficiency of data storage usage. Although response time is presented as a unified indicator of input and output measures, this is not to say that a dataset may be recorded and retrieved at the same level of frequency. The data can be pooled on a continuous basis and retrieved for analysis only for rare
validation purposes on a need basis. For such cases I altered the proposed design to accommodate the use case discussed about (here, for simplicity, the indicator is kept unified). Importantly, the designated user is prompted to assign a level of privacy. The scale allows an authorized user to select the level ranging from Confidential to Public. Additionally, access to the data repository can be restricted by a role a user plays in the system. Role-based access is constructed on the authorization procedure commonly embedded in many systems and thus provides simplest way to control the accessibility to parts of the data storage. Lastly, geographical location of access initiation can be restricted if necessary.

Second, I determine that the data storage, Audit Database component of Log4Audit, facilitates different levels of access and activity: accessible short-term, mid-term, long-term and always accessible. Always-accessible segment is not needed to be large in its size relative to the rest of the data storage. However, accessible-short-term segment plays a role of an informational dump that is necessary for validation purposes of assurance.

As a finishing step, the dashboard requires a confirmation from a user. Borrowing the Homeland Security Advisory System’s threat levels, suggested levels mirror the following indicators of security level required for the data: Ultimate, High, Elevated, Guarded, and Low.

While the defined necessary settings of Log4Audit security are designated to the identification of the level of security required for proper tier assignment in an
hierarchical data storage of Audit Database, the access to the proposed framework should also be limited on the following levels: a user, application, system and network.
Chapter 6: Observational Case Studies

“Most enterprise security is based on yesterday’s security concepts that use rules and signatures to prevent bad occurrences. What’s needed is rapid detection and response, enabled in part through behavioral analytics.”

Avivah Litan, vice president and distinguished research analyst at Gartner

In this chapter, I illustrate the decision aids built within Log4Audit framework with a help of two cases: anomaly detection and management’s assertions.

Information technology changed the decision making process in many areas of business management and audit. With heavy computerization of routine processes, decision aids evolved from manually constructed reports and graphs to software solutions that interpret the data into easily digested information. The algorithms implemented to assist the decision making process can vary in their complexity.

The range of decision aids already in use in auditing and management is wide. Bell and Carcello in their 2000 paper investigate the effects of decision aids used to assess financial statement fraud risk and find that “decision aid use has a detrimental effect on high-knowledge directors”. In the paper written by Seow in 2009, the author investigated the decision aid effect on fraud risk assessment and “how director’s identification of fraud risk factors is jointly influenced by their knowledge and the presence of generic decision aids”. Zopounidis and Doumpos (2002) study the effect of decision aids on financial decision making process and find that software packages
commonly used for decision making process need adjustments and optimization for the real-time decision making processes.

**Instant Notification Algorithm**

I propose Log4Audit framework that promotes a collection of volumes of data while unifying its search methodology and expanding its analytical depth. In my dissertation, I developed a dual engine Instant Notification Algorithm. First, I utilize the data from the application log to determine the anomalies in a set of processes. I only incorporate the internal application log to produce a set of paths with suggested level of an anomalous outcome of a process. Second, I construct the capability of the application log to provide guidance to management’s assertions. Here, I demonstrate the robustness, efficiency and consistency of the internal logging procedures by a complete match to an external set of logs.

Instant Notification Algorithm (INA) is a mathematical model. I develop this algorithm with a purpose to determine an anomaly on a sequence of events. Instant Notification Algorithm is a tool that permits an integrative collaboration with a user and thus I introduce new technology in preventive audit. My algorithm is especially successful for internal use due to its capacity to monitor real-time data provided by the Log4Audit framework.
Identification of anomalous sequence of events

I define the process mining to be an integral part of the identification of anomalous sequence of events, where the goal I set is to determine the anomaly of two strings:

\[ \text{Process}_x = (\text{event}_{x1}, \text{event}_{x2}, ..., \text{event}_{x(n-1)}, \text{event}_{xn}) \]

and

\[ \text{Process}_p = (\text{event}_{p1}, \text{event}_{p2}, ..., \text{event}_{p(n-1)}, \text{event}_{pn}) \]

where \( \text{Process}_p \) is determined as a sequence of events that constitute a proper business process, while \( \text{Process}_x \) leads to anomalous outcome of the process.

By determining the anomalies and aggregating these paths, an Instant Notification Algorithm tracks the cases within each process in real time and alerts a manager of a probability to encounter an anomaly at a completion of each process.

An anomaly is formulated with the following parameters (events):

\[ N_1(e_1), ...., N_k(e_k) \]

A process is formulated as:

\[ P_1(e_1), ...., P_m(e_m) \]

Thus, to build a set of rules that allow determination of an anomaly,
Rule = P_1(e_1), ..., P_m(e_m) => N_1(e_1), ..., N_k(e_k)   \[c,s] \]

where \( c \) is confidence of a Rule and \( s \) is its support conforming to the requirement of :

\[ P_1(e_1), ..., P_m(e_m) \cup N_1(e_1), ..., N_k(e_k) \]

and

\[ c \text{ (Rule)} = s(P_1(e_1), ..., P_m(e_m) \cup N_1(e_1), ..., N_k(e_k)) / s(P_1(e_1), ..., P_m(e_m)) \]

The confidence level of 1 here identifies a full match of a process \( P \) to an anomaly \( N \) and thus is characterized as an anomaly. The confidence level of 0 is an identifier of a normal process, not signaling an anomalous behavior of a customer. Values from 0 to 1 can be determined by a partial match of a string of events in a process matching to a set of events that are indicated as anomalous. Interestingly, meaningful segmentation of a process may lead to a rapid detection of a potentially anomalous behavior while the process has not been complete yet. However, given the focus of my dissertation (to develop a framework that assists in continuous monitoring and auditing tasks), the cases I discuss in this chapter are only an illustration.

**Guidance to management assertion**

Management’s assertions are assertive statements submitted by members of management regarding different aspects of business: transaction-level, account
balance, and presentation and disclosure. Each of the three categories are classified the following way:

- **Transaction-Level**: accuracy, classification, completeness, cutoff, occurrence;

- **Account Balance**: completeness, existence, rights and obligations, valuation;

- **Presentation and Disclosure**: accuracy, completeness, occurrence, rights and obligations, understandability.

An auditor is unlikely to start an audit prior to obtaining a letter with management assertions. A statement of management’s assertions is a good indicator of a proper process in place to produce financial statements.

I demonstrate the capacity of the application log in order to provide guidance to management’s assertions, an assertion I subjected to a formal definition:

\[ G_1(e_1), \ldots, G_m(e_m). \]

I conclude with the following statements defining the experiment with

\[ Assertion(Pos) = G_1(e_1), \ldots, G_m(e_m) \subseteq P_1(e_1), \ldots, P_n(e_n), \]

resulting in proposing an assertion, and
Assertion(Neg) = G₁(e₁), …., Gₘ(eₘ) ⊈ P₁(e₁), …., Pₙ(eₙ),

resulting in an assertion rejection.

Instant Notification Algorithm is an engine to compute a severity level of anomalies and to provide guidance on assertions to management. The uniqueness of the engine is hereditary to the framework itself — real-time results are provided for demonstrated cases of monitoring.

Although this algorithm is in its infancy, detecting only a confirmation or negation of an assertion, further development to detect a partial confidence in a proposed assertion is possible by expanding the set of segments of a process and adding possible omissions and additional steps in a process to a thread of events leading to a confirmed or rejected assertion.

**Data**

The sample size I first gathered contained 6,446 cases. All data came from one source, an organization in the event management space. 1,770 activities indicating an attempts to complete a transaction were considered for the analysis. These activities were performed within 1,550 cases. Thus, 4,896 cases I rejected due to incompleteness of the process. The process is considered complete with an indication of an attempted payment.
I determined the presence of only two processes of payment: cash and credit options available to a customer in different settings of an order submission. With further analysis, I identified 1,770 transactions and divided them into: 258 cash transactions and 1,512 credit card transactions.

My dissertation focuses on credit card transactions only\footnote{The process of credit card processing is described in Chapter 4.}. Therefore, I reject 258 transactions paid by cash. All activities’ date range is April 4th though May 3rd of 2016. Although the time span is one month, there are 458 variants of process paths. However, my further analysis of the business process and collaboration with the
management team determined that this variance is due to repetitive and cyclic actions customers take to add items to their order and are not taken into consideration as anomalous or meaningful to management’s assertions illustration case. For example, a customer who adds an item to a shopping cart then adds a member on the account and then adds another item to the shopping cart is not distinguished from a customer who adds two members to the account and adds items to each member’s name (one shopping cart for every account).

**Results**

In anomaly detection analysis, I determined that the process is very robust and does not allow any anomaly in a credit card transaction process. This finding is very powerful due to its determination of an absolute match between two independent application producing logs independently and recording the same process in different data sources. My goal was not only to verify the process, but also to determine any possible gaps in the proposed engine. The process of approving a transaction as “complete” necessitates a token to be received from a credit card processor. A transaction is marked complete when the token is received from a credit card processor stating the payment was processed, and a transaction is marked as failed when a token received indicates the status of payment as failed. Thus, the query to Log4Audit engine only indicates those transactions which, due to the procedures in place, are marked as approved or declined after the token is received from a credit card processor indicating their status. After investigating all 1,550 cases, the process I
put in place was determined as robust with no anomalies detected. The deviations in the cases were due to expected behavior of a customer.

In management’s assertions analysis, I considered general transaction related to audit objectives of completeness and accuracy. Completeness determines whether all existing transactions are recorded, and accuracy follows a verification process of recorded transaction to assert their correct amounts.

I determined completeness as verified when a transaction with a unique identifier extracted from the application database matches a transaction recorded by credit card processor. The match I considered bilaterally — credit card processor log was matched against the application data and visa versa. Absolute match I concluded through the analysis was a strong indicator of a high quality business application. All 1,512 transactions mined in the application database were identical to 1,512 transactions present in the credit card processor log. Thus, I concluded that the process in place is a strong contender to determination of all management assertions.

Both the completeness and the accuracy are dependent on a proper integration of a credit card processor and an application. Verification of accuracy of a transaction is a next step after completeness verification. The match between the values of each transaction I performed with an outcome of a 100% match.

The only issue I discovered in the process of determining the architecture of Log4Audit is of potential loss of tokens when an application server restarts and is not
available to accept tokens from the credit card processor. Therefore, each restart of an application server becomes a potential loss of data recorded in an application database. This issue was addressed and resolved by the development team prior to data collection.

**Conclusion**

The decision aids are a vital part of business operations. Log4Audit framework I offer is more than a collection of all streams of data of an organization. The collected data is indexed and ready to be mined to respond to immediate necessities of an organization and provide necessary support in making time-sensitive decisions.

The process of payment is error prone and is facilitated by many activities. However, the automation forces the processes to be independent and yet connected with the help of different tools that help matching. The capabilities of Log4Audit are strongly supported by the evidence of accuracy of data collected within Log4Audit engine.

The power of Log4Audit can be diminished by the poor quality of components where the information is entered. Although the human error in the application assessed here is impossible (all services are automated), other applications in a different business environment may cause issues with quality of Log4Audit output.
Chapter 7: Conclusions

The use of information technology in accounting and business operation is in the focus of accounting and auditing processes. Many daily business tasks became more transparent to stakeholders and auditors. With information technology deeply embedded into an organization, accounting information systems research is invested in technology capabilities, application and drawbacks. I determine the essence of continuous auditing and monitoring is in automation of business processes.

My dissertation provides a unique alternative to a common procedure for data retrieval and analysis for management and audit purposes. The Log4Audit framework I propose benefits management and internal auditors by aggregating immense amounts of data for real-time analysis and monitoring. The investigation of framework’s capabilities I offered, provides foundation to a preventive and predictive audit and monitoring. To demonstrate, I built the Log4Audit framework to collect the data and further develop the methodology for its analysis, Instant Notification Algorithm.

Unlike prior research, I extract the log files from a production environment with framework implemented prior to the collection of data. The research literature lacks analysis of log files, and rather focuses on a process mining of a structured extraction from database, or an event log. In contrast to the framework I provide, events logs mining does not allow real-time analysis and loses abundance of data
stored within an application, thus lacking the preventive component of a robust real-time monitoring of Log4Audit.

My dissertation is the first attempt to implement a well-established technique in computer science to log all the necessary information to discover the evidence of a process into accounting and audit.

With the anomaly detection and assertion recommendation, part of the Instant Notification Algorithm I propose, I demonstrate the two directionally opposite procedures: from analysis of logged data to a determination of an anomaly and a definition of an assertion in terms of events occurring within an application to a set of logged data. The reverse procedures validate the vast capabilities of the data stored within the proposed Log4Audit framework.

The full analysis of all paths of each process discovered with process mining has not been fully demonstrated in prior research. Here, I propose Log4Audit framework that permits such analysis with all determined cases. Moreover, in my work, the domain of process mining is expanded, illustrating the use of process mining technique in real-time data stream analysis.

My dissertation provides many opportunities for further research demonstrating its infancy in the research and practice domains. I determine the next step to be the development of analysis of anomalous patterns and determination of assertion recommendation patterns across the organizations and industries, as well as
investigation of an improved analysis of anomalies and development of a highly precise exception categorization in real-time monitoring.
References

ACCA, A. KPMG (2009). Accounting Sustainability Briefing: ACCAUK.


Bierstaker, J. L., Burnaby, P., & Thibodeau, J. (2001). The impact of information technology on
the audit process: an assessment of the state of the art and implications for the

Scoring and Credit Control VII*, 235-255.

Brennan, G., & Teeter, R. (2010). Aiding the Audit: Using the IT Audit as a Springboard for
Continuous Controls Monitoring. *Available at SSRN 1668743*.

California at Davis*.

3.


perspective. *IEEE Transactions on Knowledge and data Engineering, 8*(6),
866-883.

Syslog, Audit Logs, Events, Alerts and other IT'Noise'.

Coderre, D. (2015). Quantitative Key Risk Indicators: Developing Data-Driven Indicators of

Dzuranin, A. C., & Malaescu, I. (2015). The Current State and Future Direction of IT Audit:


