

ON FORMAL MODELING OF ACCOUNTING INFORMATION SYSTEMS

JAGDISH S. GANGOLLY

ABSTRACT. It is no longer possible to examine accounting information independent of the business processes that generate the data. On the other hand, accounting systems must provide information to support business processes. This paper examines the evolution of accounting systems in order to emphasize a need for formal models of accounting systems that blend models of security & integrity, models of workflows, and the traditional auditor's models of internal controls. A tentative sketch of such a theory is presented.

My experience has shown that many people find it hard to make their design ideas precise. They are . . . unwilling to express them with the precision needed to make them into patterns. . . . If you can't draw a diagram of it, it isn't a pattern. If you think you have a pattern, you must be able to draw a diagram of it. . . . A pattern defines a field of spatial relations, and it must always be possible to draw a diagram for every pattern. . . . If you can't draw it, it isn't a pattern.

Christopher Alexander (1979) in *The Timeless Way of Building*.

1. INTRODUCTION

Robert Stirling classified accounting research a long time ago into research *about* accounting and research *in* accounting. In as much as we look upon the accounting function in the society as reporting to the stakeholders on the bundle of contracts that business entities really are, as proposed by Ijiri [1971], the study of the mechanisms underlying the preparation of financial disclosures (including the various financial reports) should occupy a central role in the research *in* accounting.

Maintaining the integrity of the financial information reported, prevention & detection of corporate fraud to preserve such integrity, and ensuring that the semantics of the concepts underlying the contracts in the bundle are consonant with their measurement for reporting also should occupy a central role in the research *in* accounting. Since the contracts in the bundle are entered into with the expectation of integrity of the management as well as the measurements yielding the financial disclosures, the task of the design of accounting Information Systems that preserve the integrity also should occupy a central role in the research *in* accounting.

However, a perusal of the published literature on accounting, at least in the United States, should convince most of us that the research *about* accounting has dominated

Keynote address presented at the International Conference in Digital Accounting Research, University of Huelva (Spain), October 15, 2004. I gratefully acknowledge the assistance of Ms. Bettina Curtze for Section 3.1, which is based on an earlier joint working paper on the Computational Theory of Integrity for Accounting Systems.

THIS IS A DRAFT. PLEASE DO NOT QUOTE WITHOUT PERMISSION OF THE AUTHOR.

the discipline for the past nearly half a century. Moreover, much of the research in accounting has been motivated by economic and behavioral considerations. While the field of accounting has been enriched by such interactions, there has been a void in the research literature dealing with the analysis and design of mechanisms to facilitate provision of adequate assurance on the numbers reported. It is my contention in this paper that there is a need for unifying models for accounting *and* auditing which emphasize the behavioral aspects of information systems by *explicitly* modeling states as well as processes.

The existence of challenging problems attracts the best minds to any discipline. A prime example is the four color problem in mathematics. Other problems include Hilbert's *entscheidungsproblem*, Euler's conjectures, and Fermat's last theorem. An example of similarly challenging problems is the celebrated mind-body problem in philosophy. While it would be presumptuous for accounting to claim the same exalted status as mathematics or philosophy, in this address I hope to convince the reader that we should go beyond our quest for the low hanging fruit, and accept difficult challenges even if it means looking well beyond our discipline of accounting. The mere existence of such challenging problems is important to draw the finest minds into our discipline.

This paper is both retrospective and reflective: retrospective in that I take stock of the evolution of accounting systems, and reflective (to be more accurate, speculative) in that I try to find bridges between work in the area of secure computing (specially security modeling, and in particular the work on role-based access controls), management of workflow (in particular, workflow nets), and the analysis & design of accounting systems. In section 2 I provide a brief account of the evolution of accounting systems to set the stage for considering formal models for them. Next I briefly discuss modeling of workflows through a particular type of Petri nets called workflow nets, using a familiar purchase cycle example. Finally I provide a rough sketch of the model for an accounting system that can form a basis for a unified model to support accounting as well as auditing.

2. ACCOUNTING INFORMATION SYSTEMS

K.M. van Hee [1994] has observed that accounting systems are *monitoring information systems* in that their task is to monitor a target (or *real*) system by recording events and either signalling if such target system is in a state that satisfies certain conditions or setting off triggers to steer the target system to some desired state. Since accounting systems are *events oriented* (ie., the system state changes according to a sequence of transpired events) and the number of state changes during the life of a system is countable, they are *discrete dynamical systems*. The target system for any accounting information system is

the business operations system. The interface between the target (business operations) system and the accounting information system is provided by *sensors* (which sense the changes in the state of the target system and record them) and *actuators* (which trigger the events necessary to steer the target system to the desired state). In most accounting information systems the functions of sensors and actuators can be assigned to humans, programs, or to physical artifacts. The situation is depicted in Figure 1 below, where the interfaces between the accounting information system and the target system consist of dataflows.

The dataflows from the target system inputs and outputs to the accounting system are captured by sensors whereas the reverse dataflows are triggers which provide controls, over such inputs and outputs, either through physical devices or through signals (by way of reports) provided to humans. The processing in the target system consists of the

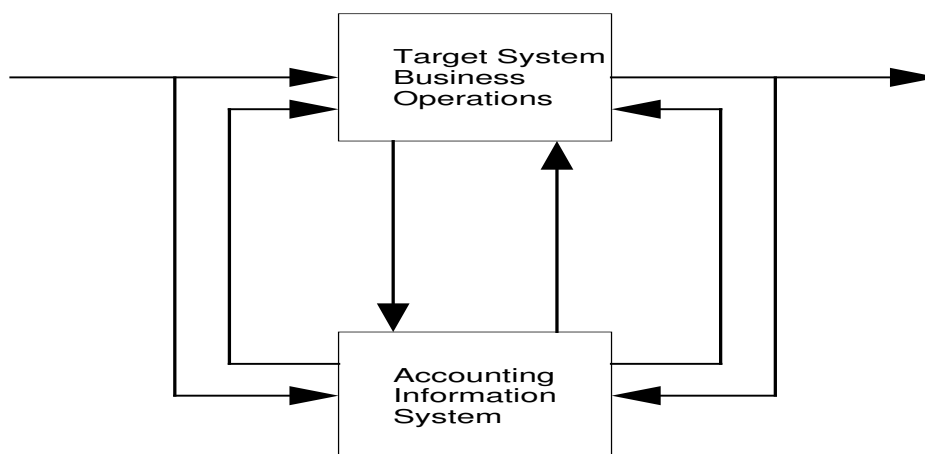


FIGURE 1. Business Operations & Accounting Information System

realization of a set of workflows in business processes. Such workflows consist of tasks (or transitions) and tokens (or, at a detailed level, data structures), and the dataflows between the target system and the accounting information system consists of the capture of such data structures in the business processes or signals to the tasks.

Traditionally, accounting information systems, as any other information systems, have been designed using either Structured systems methodologies such as SSADM, DFDs, SADT, IDEF, and ISAC, or object-oriented methodologies implemented in UML based CASE tools such as Rational Rose or Together Control Center (See for example, van der Aalst [2002]. While such tools are quite expressive and easy to use, they lack deep semantics and provide little analytical or simulation support and consequently, tools are not available to answer questions crucial to the systems analysts, posed below.

W.M.P. van der Aalst et al. [2003] state three major aggregate trends underlying recent rethinking on the way systems are designed: from programming to assembling, from data to process orientation, and from design to redesign & organic growth. I shall briefly discuss them below before turning to the evolution of accounting information systems.

2.1. From Programming to Assembling. The shift from programming to assembling was caused by at least two factors. First was the development of object-oriented programming and the development of methods (and methodologies) for object-oriented systems development. Such methods led to the development of large class libraries (such as GNU g++ class library or Hewlett-Packard's Standard Template Library for C++ language, the class libraries supporting Sun's Java J2SE or J2EE, and Microsoft's MFC class libraries). Instead of spending time and energy writing code to reinvent the wheel, systems designers and programmers could more effectively spend time and energy on developing enterprise architectures and then *assembling* (or, more realistically, *plumbing*) the code pulled off the shelves from class libraries. Second, the top-down philosophy implicit in the structured systems development methodologies that had so marvelously served systems development

proved to be ineffective where quick adaptability to environmental changes was needed. By its very nature, top-down philosophy required that the systems developer have an overall vision of the system meeting the user requirements fairly early in the systems development process. When environmental changes altered the user requirements, any change in the vision would cascade the changes throughout the design. Using object-oriented philosophy presented no such drawbacks. Interestingly, much of the work in structured systems analysis & design was inspired by the work of Christopher Alexander, a mathematician-turned architect, in architecture. However, object-oriented philosophy exploited the idea of (reusable) patterns more effectively (See Alexander [1979] quote above).

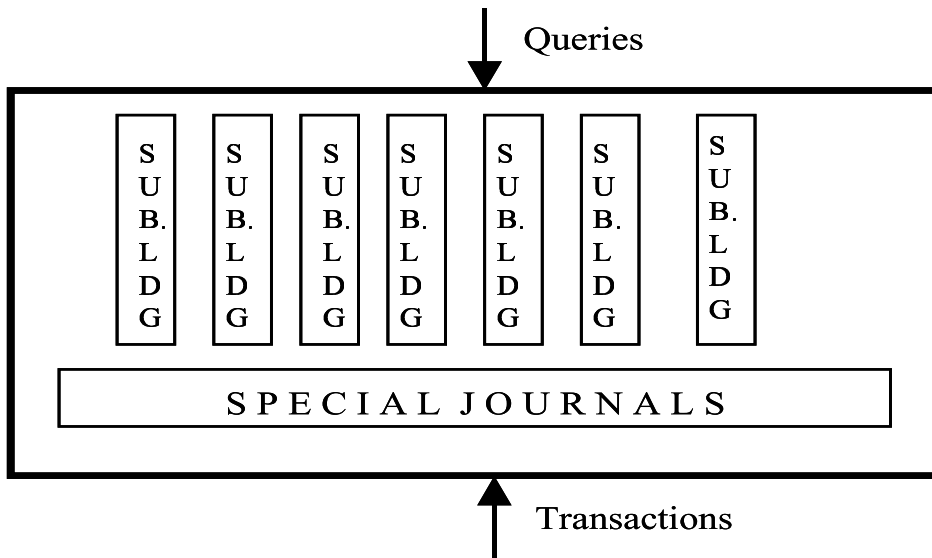


FIGURE 2. Stove-Pipe Model for Accounting Information Systems

2.2. From Data to Process Orientation. Much of the effort in the design of accounting information systems between the seventies and the mid-nineties were spent in perfecting the data-oriented way of thinking about accounting information systems. The advances in the semantic modeling of data (due to Quillian [1968] in linguistics, Chen [1976] in computing and McCarthy [1982] in accounting) were quickly absorbed by the extant literature in accounting. And so were the development of the theoretical foundations for relational databases (such as the Armstrong axioms underlying reasoning about functional dependencies, design theory leading to the normal forms, and the fundamental assumptions underlying relational databases such as the Closed-world assumption, domain closure assumption and the unique name assumption) all found wide acceptance, to a large extent because of their simplicity as well as their sound foundations in mathematics and logic. Even today, most textbooks in accounting information systems are essentially data-oriented except for a perfunctory discussion of "business processes" by way of dataflow diagrams introduced more to illustrate documentation of internal controls than to illuminate business processes in the context of accounting systems design.

Even though Petri [1962] developed the concept of Petri nets as early as in 1962, and the early work in implementing them in the context of "office information systems" was done in the mid-seventies (Zisman [1977]), as van der Aalst [2002] has observed, there were few successful applications, and the interest in Petri nets waned in the early eighties. The impetus provided subsequently by developments in computing, the ubiquity of relational databases in the corporate world, the development of the web and the consequent need to capture information from the workflows in business processes as they occur, and the popularity of *business processing re-engineering* all led to a revival of interest in process orientation in the design of information systems. Today, automation of workflows and their integration with databases is a prominent research area in information systems engineering.

In accounting, perhaps the first attempts to take a process view with explicit consideration of states, synchronization, pre- and post-conditions for processes, and related issues was in the TICOM papers (See for example Bailey et al. [1985]), followed by the dissertation of Verghese [1988] at Columbia, but sadly their efforts were largely ignored in the literature. The work of Hamscher [1995] and Natovich and Vasarhelyi [1998] dealing with AI aspects of control in accounting systems also had elements of process orientation, but that strand of work also was not followed through in the literature. One can speculate reasons for such neglect, but I suspect the reason to be the then lack of automation of business processes and the abundance of paper evidence for the auditors to pour over. It is only recently that the study of business processes in accounting has received any attention, and even the scant attention it has received has been primarily in the informal study of internal controls in auditing.

2.3. From design to redesign & organic growth. The tremendous growth in the use of the internet has made adaptability to frequent changes in the systems a necessity. Instead of building a system based on a grand design from the scratch, it is necessary to build systems from off-the-shelf components in such a way that frequent changes can be made on-the-fly.

2.4. Evolution of Accounting Information Systems. In order to understand the significance of this move from data to process orientation let us look at the developments since the mid-sixties in relation to our reference model for accounting information systems in Figure 1. Prior to computerization, accounting systems in most medium-sized and large corporations were stove-piped in design and rather flat in architecture as shown in Figure 2. The design mirrors how elementary financial accounting is taught in most universities in the United States even today; accounting data is sequestered from rest of the enterprise data (at least conceptually). The transactions are entered in the special journals and the queries are analyzed through the subsidiary ledgers.

In the early days of computerized accounting systems, this stove-pipe model was implemented on top of rudimentary operating systems of the day and the subsidiary ledgers were subsumed in the application stove-pipes, leading to what can be termed Babelization of accounting data; the data structures as well as data semantics were proprietary to the individual stove-pipes. Data sharing across applications was virtually impossible and so were enterprise-wide data semantics.

The introduction of database systems dramatically transformed accounting information systems by the introduction of common enterprise-wide data models. Such enterprise-wide

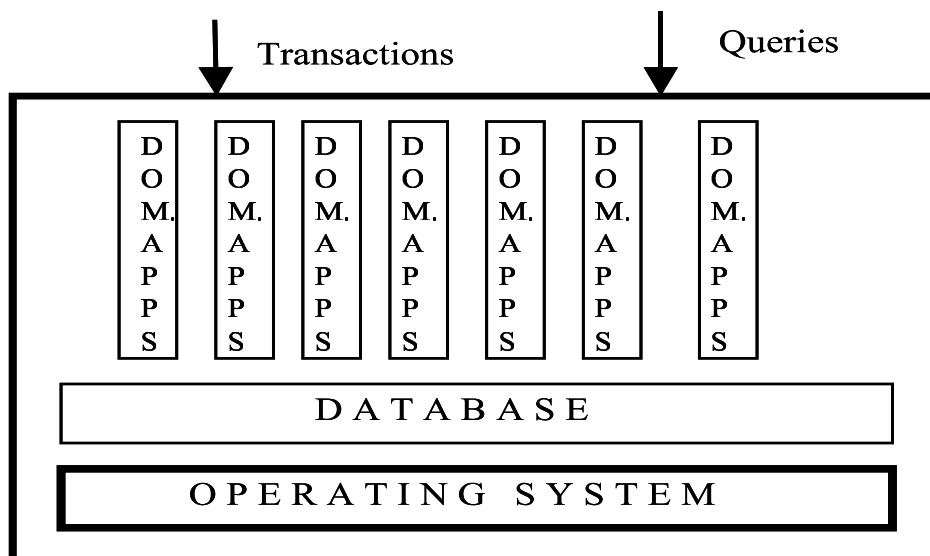


FIGURE 3. Accounting Information Systems Based on Databases

data models led to the integration of accounting as well as non-accounting operational data into a single model. This is shown in Figure 3.

The database and the domain applications consisting of database query language (such as SQL) statements, wrapped in an imperative language such as COBOL or C, subsumed the traditional accounting data. While the database was enterprise-wide, the operational and accounting systems were still perceived by accountants as two different things with interfaces, and therefore the workflows in accounting were also perceived as different, if not independent, from the operational workflows. Even today, in most accounting information systems texts we teach systems flowcharts and dataflow diagrams while covering operational workflows only to the extent we can not avoid them in the discussions of internal controls. Control flows, the staple of systems engineers, on the other hand, are just about totally ignored.

This should come as no surprise considering our traditional preoccupation with financial data in the context of financial statement preparation and disclosures. Perhaps the only striking exception to this rather parochial data-orientation is the seminal work in the area of REA accounting by McCarthy [1982]. The introduction of internal as well as external agents in the data model may seem rather awkward, but its significance can not be overstated. Zachman [1987], in his seminal work on enterprise architectures states the necessity of information systems being informed by answers to the Kiplingeresque questions why, what, how, who, when, and where. McCarthy's REA model for the first time introduced answers to the question who in an essential way in data modeling, and coincidentally forced us to look to workflows in the business processes.

After the databases, perhaps the most important innovation in information systems design, though bypassed by most of the accounting information systems literature to date, is the automation of workflows. Initially proposed in the computing literature as "office information systems", in my opinion, in the future it will have a dramatic impact on the way accounting information systems are perceived, modeled, designed, implemented, and

audited. Figure 4 provides a sketch of the architecture where workflow and ERP systems (which usually have a workflows management component).

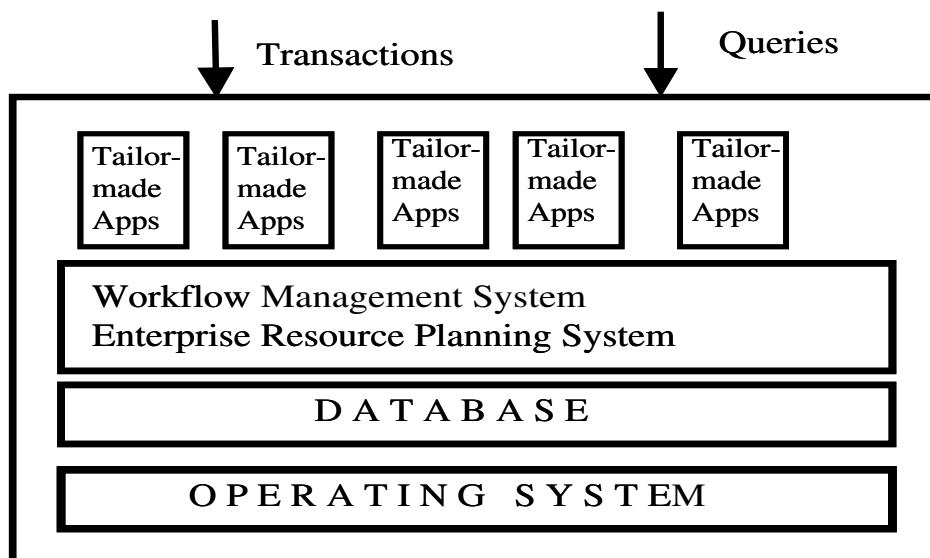


FIGURE 4. Accounting Information Systems Based on Workflow/ERP based systems

To summarize, accounting systems have evolved from simple double-entry book-keeping systems into an integral part of enterprise information systems built around the workflows through which business operations are conducted. With the workflows forming the most important interface with the database on which corporate information resides, we in accounting will need to reconfigure our models so that capture of the accounting data is at the workflow level. In the next section I shall provide the motivation for formal modeling of accounting systems and the study of its computational properties.

3. ON MODELING ACCOUNTING INFORMATION SYSTEM

Building formal (mathematical) models of accounting information systems is of paramount of importance for at least three reasons: to study its properties, to facilitate planning, and provide a robust theoretical foundation for accounting as well as auditing.

First, the model of an accounting information system can enable us to study its properties *before* it is implemented. It must help us answer questions such as: What are its *control* properties? How *resilient* is it to various inputs? How does it *scale* with respect to the computational burden, ie., what is the *computational complexity* of the algorithms underlying the system? Is it *correct* in that it does not have logical flaws which will result in anomalies later on? Is it *efficient* in that alternative implementations of the system do not yield superior results? Does it *behave* in the way the user expected it to behave?

Second, such a model can aid planning, through simulations, by facilitating answers to questions such as: Will the present computing capacity suffice should the transaction load

grow by a specified factor? How much computing capacity will be needed for a certain transaction density? Of course the answers to these questions will depend on the way the human resources are deployed. Nonetheless, it is possible to answer the questions in the abstract by making assumptions regarding such deployments.

Third, such a model can provide a theoretical foundation for the discipline of auditing. Here, we are looking for answers to questions such as: Does the *system* have controls *adequate* in a specified sense? What is the *computational burden* of answering such a question? Assuming a certain transaction density, what is the computational burden of an algorithm to test the database underlying such a system to provide assurance that the controls operate as specified in the model?

The questions I have raised above are quite important if accounting information systems is to thrive as a discipline. Interestingly, they are also questions that have been asked in related disciplines, specially computational science (questions relating to complexity), and systems engineering (questions relating to control properties, correctness, efficiency, verification and validation). In the remainder of this section, I will provide a small example of modeling of workflows, and then provide a framework for examining the questions that I have raised above.

3.1. Business Processes as Workflow Nets. Auditors have documented business processes in a number of ways including systems flowcharts, and logical & physical dataflow diagrams. Systems engineers also have used formalisms such as control flow diagrams (See Hatley and Pirbhai [1987], Mills and Gomaa [2003], Tourlas [2001]), statecharts (Harel [1987], Stolzenburg [2001]), and more recently as Petri Nets of the kind referred to as workflow nets (Knorr [2000], Knorr and Stormer [2001], Knorr and Weidner [2001], Aalst et al. [1994], Aalst [1994], Aalst and Hee [1996], Aalst [1998], Aalst et al. [1994], Aalst and Hee [1995]). In this paper I will model business processes as such workflow nets to represent business transactions from an accounting standpoint. Clark & Wilson (1989) refer to such choreography of business processes as 'well-formed transactions'. Such a formalization provides a formal and yet graphical way of representing information processing systems that are 'concurrent, asynchronous, distributed, parallel, nondeterministic, and/or stochastic' (See Murata [1989] - characteristics that apply very well to any accounting system).

There are many good reasons to consider workflow nets as the formalism for representing business processes. As Aalst (1996) has observed, they provide formal semantics in spite of their graphical nature. Secondly, unlike representations such as dataflow diagrams which model events explicitly but states implicitly, workflow nets are state-based where tasks are modeled as *transitions* and intermediate states are explicitly modeled as *places* (as our example below will show). Thirdly, in using Petri Nets the system designer has at the disposal an abundance of analysis techniques. Fourthly, as Aalst (1994) has observed, workflow management systems underlying the business processes are similar to operating systems where Petri Nets have been fruitfully applied. In fact, it not an exaggeration to say that operating systems are, in a sense, accounting systems too (or conversely, accounting systems are operating systems too).

I will motivate the discussion of *workflow net* representation of business processes with a purchase/accounts payable example with which we all are familiar. Workflow net is a kind of *Petri net*. In a Petri Net, there are two types of nodes, called *places* and *transitions*. Places denote documents (or data) and are represented by circles. Transitions are *tasks* (done by people, programs, or physical devices) in the business process, which are denoted by rectangles. The directed edges (or arcs) in Petri Nets denote the *flow of work* in the business process, and are denoted by arrows, the direction of the arrows representing the direction of workflow. A workflow net is a Petri net that satisfies the following two

properties: 1. There are two special places say, P1 (source) and P14 (sink) such that P1 is not preceded by any transition and P14 is not followed by any transition, and 2. If we add a transition t^* and two arrows from P14 to t^* and from t^* to P1, then the resulting Petrinet is strongly connected in that there is a path (a sequence of arcs) between any two nodes. Here, I have used the notation in Figure 5 below.

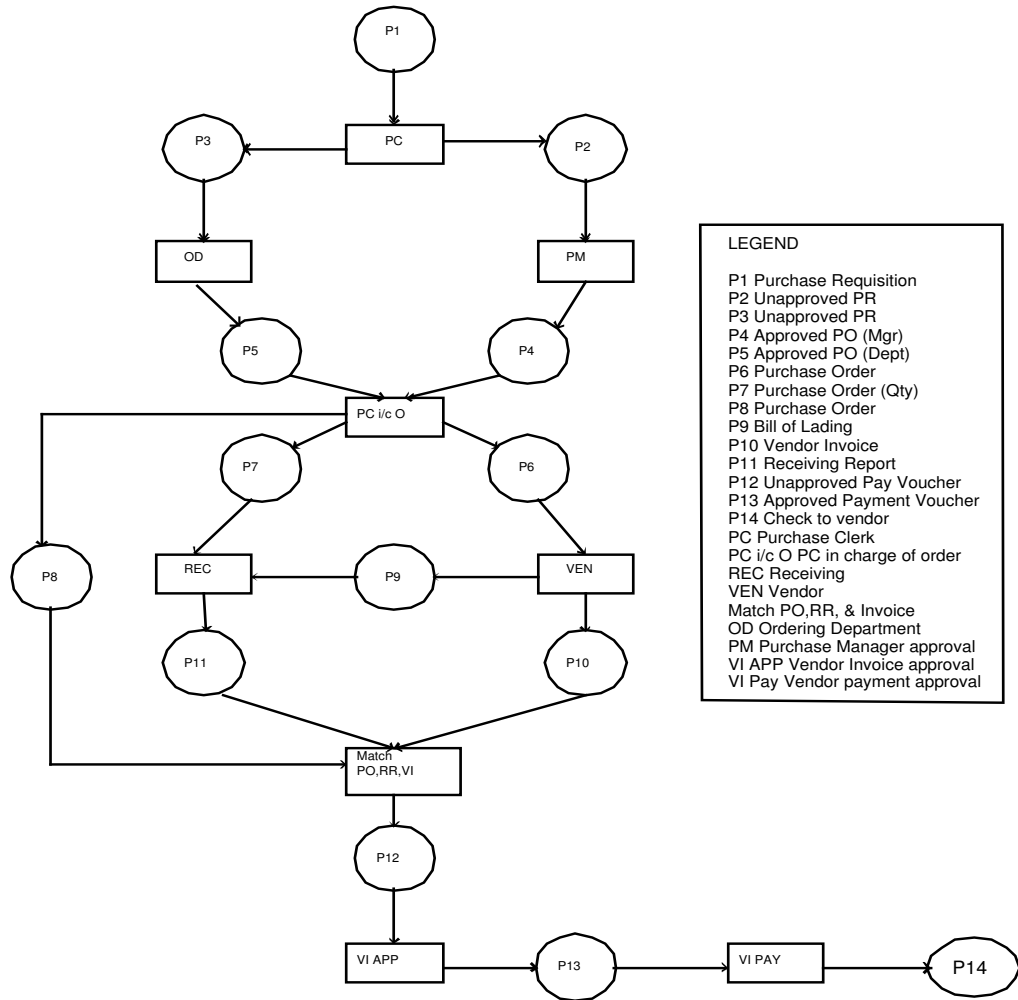


FIGURE 5. A Simplified Workflow Net for a Typical Purchase Cycle

3.1.1. *A Purchase/Accounts Payable Example:* Consider an organization where a purchase requisition (P1) is prepared by a department needing an item. We take the purchase requisition as triggering the process. When it is received by a purchase clerk, (s)he obtains quotation from approved vendors, and transmits such unapproved purchase requisitions (with quotations) for approval by the purchase manager (PM) and the ordering department clerk (OD). Such Purchase requisitions approved by the ordering department (P5) and by the Purchase Manager (P6) are received by the purchase clerk in-charge of ordering (PC i/c O) who transmits the Purchase order (P6) to the vendor (VEN), a copy of the purchase order without order information (P7) to the Receiving clerk (REC), and another copy of the purchase order to the Accounts Payable clerk (Match PO, RR, VI).

When the goods are received by the Receiving Department along with the Bill of Lading (P9), the receiving clerk prepares the Receiving Report (P11), which is transmitted, to the Accounts Payable clerk. When the Vendor Invoice (P10) is received by the Accounts Payable clerk, (s)he will match the Vendor Invoice with the Purchase Order and the Receiving Report and prepare a Payment Voucher (P12) for approval to the Manager who is authorized to approve Vendor Invoices (VI APP). The approved Payment Voucher (P13) is transmitted to the Payments clerk (VI PAY).

Figure 5 shows the workflow net for the above example. For the sake of readability, I have labeled the transitions by the employee performing the function rather than the function itself. In most accounting systems, however, the transitions are tasks which are assigned to roles occupied by specific employees. It is important to note that the places in Figure 5 represent data structures, which may be relations or objects. For example, P1 (Purchase requisition) is a relation or an object consisting of data on the requesting department, the requesting person, authorized by, items and quantities, and so on. Likewise, P2 (Unapproved Purchase Requisition sent to the Purchase Manager) also is a relation or an object consisting of information on the requesting department, the requesting person, authorized by, items and quantities, the vendor, unit prices, and so on.

Each purchase requisition is a tuple if P1 is implemented as a relation, or an object if implemented in an object-oriented framework. When the transition PC 'fires', this tuple (or object) is 'removed' (deleted) and the necessary information in it is 'inserted' into a corresponding tuple in P2 (or an object created in P2). Ordinary workflow nets use *tokens* to represent the states of the network. The networks where the places are data structures are also sometimes called *Predicate/Transition networks* (For details, see Oberweis and Sander [1996]).

It should be obvious from the above example that the workflow nets enable the designers of accounting information systems to unify the enterprise models by the interfaces between tasks in the operational system and the data in the accounting system provided by the arcs between the transitions (tasks) and the places (data structures) respectively. Since the workflow nets provide templates for various business processes, the workflow net framework also provides a rigorous and verifiable way of auditing controls over such processes as well as their implementation in individual cases. It is possible to answer questions such as: is the choreography of the business processes meet the behavioral expectations of the user (and designer)? Is such choreography followed in individual cases (or are there deviations from the norms)? Such questions are always asked by the auditors in the context of evaluation of the internal controls in accounting systems.

Now we are ready to build a process-oriented formal model for accounting information systems that models states explicitly. In the next section I will speculate on a formal model for an accounting system.

3.2. Towards a Formal Model of an Accounting Information System. Attempts to axiomatization of accounting systems date back from the early work of Mattesich (See also Mattesich [1978]) and Ijiri [1983] (See also Ijiri [1971]), but their work did not deal explicitly with modeling of tasks and states because description of the system was not their primary objective. Moreover, their work was not motivated by the need for integrating operational and accounting systems by models that could be verified, validated, and support hardware/software/human resources aspects of planning as well as auditing. In this section I will provide a rough sketch of a formal model for accounting information systems. A more detailed treatment of the model including the representation of accounting data structures must wait another paper¹

Consider an enterprise consisting of a set of persons P and a set of business processes \mathfrak{B} . A generic business process $B \in \mathfrak{B}$ can be represented as a workflow net, such as the one in Figure 5 above, given by $B = \{T, D, F\}$ where T is a set of tasks, D a set of predicates (data structures), and $F \subseteq (D \times T) \cup (T \times D)$ a set of flow relations between tasks and predicates. For example, for the workflow net in Figure 5, we have

$$\begin{aligned} T &= \{PC, OD, PM, PC \text{ i/c } O, REV, VEN, \text{Match PO/RR/VI}, VI \text{ APP}, VI \text{ PAY}\} \\ D &= \{P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14\} \\ F &= \{(P1,PC), (PC,P3), (PC,P2),(P3,OD), (P2,PM), (OD,P5),(PM,P4), (P5,PC \text{ i/c } O), \\ & (P4,PC \text{ i/c } O), (PC \text{ i/c } O,P6), (PC \text{ i/c } O,P7), (PC \text{ i/c } O,P8), (P7,REC), (P6,VEN), \\ & (VEN,P9), (P9,REC), (REC,P11), (VEN,P10), (P9,\text{Match PO/RR/VI}), (P10,\text{Match PO/RR/VI}), \\ & (P11,\text{Match PO/RR/VI}), (\text{Match PO/RR/VI},P12), (P12,VI \text{ APP}), (VI \text{ APP}, P13), (P13,VI \text{ PAY}), \\ & (VI \text{ PAY},P14)\} \end{aligned}$$

The set of tasks T for a business process $B \in \mathfrak{B}$ in the enterprise can be partitioned into sets T^A , T^E , T^C and T^R denoting the sets of authorization, execution, custody, and recording task modes respectively as we do in auditing. I will denote by $T^{\bar{A}}$ the set complement of T^A in T . For example, $T^{\bar{E}}$ is the set of all transactions in B except execution transactions, or $T - T^E$. This is consonant with our understanding of internal control in enterprises.

The tasks in T are performed by personnel in P . More specifically, each task is performed by a person by assuming a specific role. For example, in Figure 5 purchase clerk and purchase manager are roles. Suppose Bob and Alice are purchase clerks (more accurately they are authorized to perform the role *purchase clerk*) while Arthur and Betty are purchase managers (more accurately they are authorized to perform the role *purchase manager*).

The concept of roles is intimately related to the concept of multi-level security in the Orangebook (See *Trusted Computer System Evaluation Criteria* (TCSEC) in of Defense [DOD]), and the work in Role Based Access Control (See Sandhu [1988, 1990], Ahn et al. [2000], Sandhu et al. [2000], and Kandala and Sandhu [2001]). We can think of roles in an enterprise similarly arranged as a lattice (or partial order) consonant with the organizational hierarchy, represented as a partial order as is done in most multi-level security models (See for example, Bell and La Padula [1975], Bell and La Padula [1996], Bishop [2003], Clark and Wilson [1987], Gollmann [2001]). One consequence of this is that while each person in the enterprise has a role assigned to him/her, such a person may be able to contingently assume some other lower level role in specific transactions. For example, in Figure 5, a purchase manager may be authorized to assume a lower level role (say, that of a purchase clerk) for a particular transaction (case) on-the-fly; when both Bob and Alice

¹Here I shall not provide the background on workflow models and security models forming the basis. Those interested may like to read the references at the end of the paper.

call in sick, Arthur may assume the role of a purchase clerk in processing the purchase requisition for a dry felt from Tom in the Machine house.

The enterprise must assign to each person the roles he/she can assume, and also assign to tasks the roles that a person performing the task should have authorization for. The assignment of roles to personnel determines what one can do, and the assignment of roles to tasks determines what authority a person must have to perform them. Thus we have role assignment mapping $f_{PR} : P \rightarrow 2^R$, and the task assignment mapping $f_{TR} : T \rightarrow 2^R$. Since each task must be assigned to a person in the organization, I will assume that for each $\tau \in T$, $f_{PR}(p) \cap f_{TR}(\tau) \neq \phi$ for some $p \in P$.

An accounting information system also has a set of control rules \mathfrak{R} . The control rules include those on separation of duties (between execution, custody, and recording), rules derived from organization policies, and rules governing business practices. Such rules can be represented as Horn clauses as done in Knorr and Weidner [2001]. Such control rules are applied to tasks in the business processes based on the history of tasks completed till the moment that the rule is to be applied. Assuming that there are no loops in the workflow net, we have a partial order (T, \leq) on the set of tasks. If the tasks are indexed, then for $\tau_i, \tau_j \in T$, if $\tau_i \leq \tau_j$, then the internal control rules can be applied to the history of the business process B up to the time of firing of the task τ_j , given by $\{(\tau_i, p_i), (\tau_{i+1}, p_{i+1}), \dots, (\tau_{j-1}, p_{j-1})\}$, sometimes referred to as an execution chain.

An example of a control rule for our Figure 5 illustration might include, the rule that if in a transaction a person assumes the role of a purchase clerk to perform the task of transmitting the purchase requisition, then he can not perform any other task in the transaction that belongs to the authorization, recording, and custody partitions. Denoting the performance of any task τ_i by $p_k \in P$ as $p_k(\tau_i)$, we can state this control rule as follows: For any two tasks $\tau_i^m, \tau_j^n \in T$ such that $\tau_i \leq \tau_j$, $(p_k(\tau_i^m) \wedge p_k(\tau_j^n)) \longrightarrow m = n$.

Now we can define an accounting information system as $\Sigma = \{P, \mathfrak{B}, \mathfrak{R}, f_{PR}, f_{TR}\}$. An instance of an accounting information system is given by the accounting system and a sequence of execution chains for the business processes during a period. It should be obvious that the question if all control rules are followed is answered by computing that the tasks in the execution chains for all the business processes follow the templates given by their corresponding workflow nets, that the task-role and person-role mappings do not violate the rule on segregation of functions, and that the various control rules have been observed by the execution chains of all business processes. The above is a rough sketch of a formal

model that can provide resilient foundation for both accounting information systems and auditing. Work is under way at Albany to build the model, develop algorithms, and study their properties.

4. CONCLUDING OBSERVATIONS

We have moved from a world where accounting system was sequestered from the rest of the enterprise information systems and physical access controls were exercised over accounting records. With the pervasiveness of database systems and enterprise-wide integrated information systems, accounting information systems will be subsumed within such enterprise systems the same way that monitoring systems are seamlessly integrated within physical systems. This provides excellent opportunities for research in interdisciplinary formal modeling where results in secure computation & information assurance, workflow modeling, and business process re-engineering are brought to bear on the design of accounting information systems.

REFERENCES

- W. v. d. Aalst. Putting petri nets to work in industry. *Computers in Industry*, 25(1): 45–54, 1994.
- W. v. d. Aalst. The application of petri nets to workflow management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.
- W. v. d. Aalst and K. v. Hee. Framework for business process redesign. In J. Callahan, editor, *Proceedings of the Fourth Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 95)*, pages 36–45. IEEE Computer Society Press, Berkeley Springs, April 1995.
- W. v. d. Aalst and K. v. Hee. Business process redesign: A petri-net-based approach. *Computers in Industry*, 29(1-2):15–26, 1996.
- W. v. d. Aalst, K. v. Hee, and G. Houben. Modelling workflow management systems with high-level petri nets. In G. D. Michelis, C. Ellis, and G. Memmi, editors, *Proceedings of the second Workshop on Computer-Supported Cooperative Work, Petri nets and related formalisms*, pages 31–50, 1994.
- G. Ahn, R. Sandhu, H. Myong, and J. Park. Injecting rbac to secure a web-based workflow system. In *Fifth ACM Workshop on RBAC*, 2000.
- C. Alexander. *The Timeless Way of Building*. Oxford University Press, 1979.
- A. Bailey, G. Duke, J. Gerlach, M. R. KO, Chen-En, and A. Whinston. Ticom and the analysis of internal controls. *Accounting Review*, LX(2), 1985.
- D. E. Bell and L. J. La Padula. Secure computer systems: Unified exposition and multics interpretation. 1975.
- D. E. Bell and L. J. La Padula. Secure computer systems: Mathematical foundations, volume i. November 1996.
- M. Bishop. *Computer Security: Art and Science*. Addison-Wesley, 2003.
- P. P. Chen. The entity relationship model: Toward a unified view of data. *ACM Transactions on Database Systems*, pages 9–36, 1976.
- D. Clark and D. Wilson. A comparison of commercial and military security policies. In *Proceedings of the 1987 IEEE Symposium on Security and Privacy*, pages 184–194, 1987.
- D. Gollmann. *Computer Security*. John Wiley, 2001.
- W. Hamscher. New horizons in audit technology. In *Proceedings of EDPAC-95*, Melbourne, Australia, 1995.
- D. Harel. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8:231–274, 1987.
- D. Hatley and I. Pirbhai. *Strategies for Real-Time System Specification*. Dorset House, New York, 1987.
- Y. Ijiri. *Accounting in Perspective : Contributions to Accounting Thought by Other Disciplines, R . Sterling and W . Bentz (eds)*, chapter Logic and sanctions in accounting. Houston : Scholars Book Co., 1971.
- Y. Ijiri. On the accountability based conceptual framework of accounting. *Journal of Accounting and Public Policy*, 2:75–81, 1983.
- S. Kandala and R. Sandhu. Secure role-based workflow models. 2001.
- K.M. van Hee. *Information Systems Engineering: A Formal Approach*. Cambridge University Press, 1994.
- K. Knorr. Dynamic access control through petri net workflows. In *Proceedings of the 16th Annual Computer Security Applications Conference (ACSAC)*, pages 159–167, New Orleans, Dec 2000.
- K. Knorr and H. Stormer. Modeling and analyzing separation of duties in workflow environments. In *Proceedings of 16th International Conference on Information Security (IFIP/Sec)*, pages 199–212, Paris, France, June 2001.

- K. Knorr and H. Weidner. Analyzing separation of duties in petri net workflows. In *Proceedings of the International Workshop on Mathematical Methods, Models and Architectures for Computer Networks Security*, pages 102–114, St. Petersburg, Russia, May 2001.
- R. Mattesich. Towards a general and axiomatic foundation of accountancy , with an introduction to the matrix formulation of accounting systems. *Accounting Research*.
- R. Mattesich. *Instrumental Reasoning and Systems Methodology*. Dordrecht : D . Reidel, 1978.
- W. E. McCarthy. The rea accounting model: A generalized framework for accounting systems. *Accounting Review*, pages 554–577, 1982.
- K. L. Mills and H. Gomaa. A knowledge-based method for inferring semantic concepts from graphical models of real-time systems. May 2003.
- T. Murata. Petri net theory and the modeling of systems. *Proceedings of the IEEE*, 77 (4):541–580, 1989.
- J. Natovich and M. A. Vasarhelyi. Business process modeling from the control perspective: The ai planning approach. *International Journal of Intelligent Systems in Accounting, Finance & Management*, 6(2):121–139, 1998.
- A. Oberweis and P. Sander. Information system behavior specification by high-level petri nets. *ACM Transactions in Information Systems*, 14(4):380–420, 1996.
- D. of Defense (DOD). *Trusted Computer System Evaluation Criteria*, dod 5200.28-std edition, 1985.
- C. Petri. *Kommunikation mit Automaten*. Phd thesis, Institut fr instrumentelle Mathematik, Bonn, 1962.
- M. R. Quillian. *Semantic Information Processing*, chapter Semantic Memory, pages 227–270. MIT Press, 1968.
- R. Sandhu. Transaction control expressions for separation of duties. In *Proceedings of the Fourth Aerospace Computer Security Applications Conference*, pages 282–286, 1988.
- R. Sandhu. Separation of duties in computerized information systems. In *Proceedings of the IFIP WG 11.3 Workshop on Database Security*, 1990.
- R. Sandhu, D. Ferraiolo, and R. Kuhn. The nist model for role - based access control: Towards a unified standard. In *Fifth ACM Workshop on RBAC*, pages 47–64, 2000.
- F. Stolzenburg. From the specification of multiagent systems by statecharts to their formal analysis by model checking. 2001.
- K. Tourlas. *Diagrammatic Representations in Domain-Specific Languages*. Ph.d dissertation, University of Edinburgh, 2001.
- W. van der Aalst. Making work flow: On the application of petri nets to business process management. In J. Esparza and C. Lakos, editors, *ICATPN*, volume 2360 of *Lecture Notes in Computer Science*, page 122. Springer-Verlag Berlin Heidelberg 2002, 2002.
- T. Verghese. *A Formalization of Internal Control Evaluation*. PhD thesis, Columbia University, 1988.
- W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G.Schimm, and A.J.M.M. Weijters. Workflow mining: A survey of issues and approaches. *Data & Knowledge Engineering*, 2003.
- J. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3), 1987.
- M. Zisman. *Representation, Specification and Automation of Office Procedures*. Phd thesis, University of Pennsylvania, Warton School of Business, 1977.