Structural Artifacts in Method Engineering: The Security Imperative

Richard Baskerville
Copenhagen Business School and
Binghamton University
Binghamton, New York 13902 USA
Tel +1 607 777 2337 Fax +1 607 777 4422 Email baskerville@cbs.dk

Abstract
The organizational structure has to do with human relationships, and is distinguished from the various artifacts (like information technology, systems development methods, and other mechanical products) that reflect those relationships. Information technology represents a first-level artifact and systems development methods represent a second-level artifact. This paper explains and illustrates a theory in which method engineering introduces third-level structural artifacts in organizations. A demonstration is included that uses security as one of the system imperatives that must be captured by third-level structural artifacts such as method engineering. This demonstration shows how method engineering may produce methods that are more complete and more harmonized with the organizational situation.

Keywords
Information Systems Development, Systems Development Methods, Software Engineering, Organizational Structure, Information Systems Security

1 INTRODUCTION

There are a large number of widely varied methods available for information systems developers. These include structured approaches (*e.g.,* Yourdon 1989), prototyping approaches (*e.g.,* Connell and Shafer 1989), information engineering (*e.g.* Finkelstein 1989), soft systems (*e.g.,* Checkland and Scholes 1990), sociotechnical (*e.g.,* Mumford 1983), object-oriented (*e.g.* Embley, Kurtz and Woodfield 1992), *etc.* Many of these methods have been comparatively analyzed in books (*e.g,* Olle *et al.* 1988 or Avison and Fitzgerald), and journal articles (*e.g.* Jackson and Keys 1984, Jayaratna 1988 or Hirschheim and Klein 1992). Despite a fairly large body of work concerning the details of systems development methods, there is still a very poor understanding of how such methods are actually used in practice (Wynekoop and Russo 1993) or even whether these are ever used at all (Baskerville Travis and Truex 1992).

Method engineering (Kumar and Welke 1992) represents the effort to improve the usefulness of systems development methods by creating an adaptation framework whereby methods are created to match specific organizational situations. The goals of this adaptation framework include at least two possible objectives. The first objective is the production of contingency methods, that is, situation-specific methods for certain types of bounded organizational settings. This objective represents method engineering as the creation of a multiple choice setting. For example, in a systems consulting-firm situation, method engineering might be used to create a number of alternative predetermined methods, and each new client's situation might be analyzed to select one of the methods which would be most appropriate for use. The second objective is one in which method engineering is used to produce methods "on-the-fly". Each systems development project begins with a method definition phase where the development method is invented on the spot. In this second objective, method engineering is a mechanism for coping with the uniqueness of each development setting. Organizational change is involved because it contributes to this uniqueness. The mechanism operates by lifting the systems structures to a higher (third) level of abstraction, such that the actual development structures become "selectable" (or definable), and importantly, the determination of these selections itself becomes more highly structured.

The purpose of this paper is to explain and illustrate a theory of method engineering which is oriented toward these third-level structural artifacts in organizations. Third-level artifacts represent to the imperatives of the method engineer. This purpose is addressed by four major sections. In the remainder of this first section, we will define several key terms. The second section will analyze the relationship between information systems and organizational structures in terms of structural artifacts. The third section extends this analysis to the new artifacts demanded by the new level of abstraction introduced by method engineering. Following this, the fourth section illustrates the rather positive nature of these new artifacts using information systems security as an example. The final section summarizes the demonstration and discusses some research issues that are opened by the analysis.

It is not the purpose of this paper to directly propose method engineering techniques and structures like tool selection heuristics, notation inventories or analytical techniques directed toward the target organizational situation. The paper will not attempt to survey the various imperatives to which method engineers must respond. Rather, an analysis is presented which can frame a better understanding of how such proposals will interact with human organizations, information systems, and system development methods. However, the analysis is illustrated using an outline of possible security notation and criteria which would be appropriate in method engineering.

For the purposes of this paper, the term information technology (IT) will suggest a broader view than "just computers", including telecommunications and office technologies like photocopiers. Also IT is not bound to machinery, but includes conceptually-grouped technologies (*e.g.*, object-oriented or prototyping concepts). It is arguable whether definitions of information systems and information technology may encompass each other. In this paper, these are separate but closely related concepts, information systems (IS) refers to the systematic development, operation and management of IT as well as the IT itself. We are especially concerned with this "systematic development" component in IS, and we will use the term information systems development (ISD) to refer to the analysis, design and implementation components embedded in our definition of IS.

Both Oxford and Webster's dictionaries primarily define the term "*method*" as meaning "the procedure for obtaining an object." The secondary definitions fasten on such ideas as "orderly," "systematic," "regularity," and "regimen." Method is clearly a concept of process rather than representation. This paper will avoid the term *methodology* altogether, for in the field of IS, the original meaning of this term (the study of method) has become confused, and is either used as a simple synonym for method (*cf.* Olle *et al.* 1988, p. 1) or to create a hierarchy of methods (*cf.* Jayaratna 1993, Wynekoop and Russo 1993) which has been shown to be rather strained when closely examined: It is a higher-order version of the same construct: "a method of methods" (Oliga 1988, p. 90)

The concept of "artifact" is especially important in this work. An artifact is an object made by people, usually with skill, for subsequent use. Its common archeological and anthropological usage also implies that the object is from an earlier time or cultural stage. This implies that an artifact has a physical persistence These connotations are important, because these distinguish the making of the object from its use, imply that such objects are cultural icons, and that their existence may endure through later periods of time and cultural stages.

Information systems security is used to illustrate the points below regarding IS organizational artifacts. For this paper, "security" is defined broadly to include not only features that prevent intentional losses, such as fraud and vandalism, but also unintentional losses such as natural disasters and errors. Thus security encompasses system integrity and reliability. Security is presented as one "imperative" of systems development methods. Such imperatives are fundamental goals of the systems development that motivate the inclusion of certain absolutely-necessary features into a method's design. For example, imperatives like maintainability or reusability motivate features like encapsulation or inheritance in an object-oriented method.

2. STRUCTURAL ARTIFACTS OF ORGANIZATIONS

The information system has an important relationship with organizational structure. The development methods are also directly or indirectly elements of this relationship. This implies that there is also a relationship between method engineering and organizational structure.

There is a clear distinction between the structure in a human organization and the artifacts which reflect that structure. It is possible for the human organization to conflict with its structural artifacts. For example, in many organizations the CEO's secretary wields real power, like autonomously making decisions in assigning responsibilities further "down the line". Every person in the organization will be aware of this line authority, yet it almost never appears in the organizational chart or position descriptions. The human organization differs in reality from the artifacts that supposedly define it.

This important distinction between the structure of the human organization and the artifacts intended to reflect that structure requires a precise terminology. The terms "organization", "structure" and "system" will appear below as icons for fairly strict dictionary concepts. *"Organizations"* is a term that regards people who are dividing their work together for some common purpose. *"Organized"* is a term that regards something formed into a whole consisting of interdependent or coordinated parts especially for united action. Organization is defined recursively: A group of persons or smaller organizations organized for some end or work. This paper will use the term organizational structure to regard the persistent relationships between the people (or smaller organizations) in organizations. "Persistent" regards repeated instances of relationships that occur with regularity.[1]

2.1 IT And Other Structural Artifacts

There are widely varied viewpoints about the relationship between IT and organizational structure. However, many of these seem to assume that the information technology is somehow an elemental part of the organizational structure. (Five of these viewpoints are surveyed in the appendix to illustrate how each relates to this assumption.) However, organizational structure regards persistent *relationships between people*. While the IT might enable or reflect this relationship, it does not embody the relationship itself. It is important to carefully distinguish organizational structure (human relationships) from the IT artifacts (mechanical products) that reflect those relationships. (In this point we are carefully distinguishing between IT, which is artifactual in our definition, and IS which may be seen as a "web" or "institution" that is inherently embedded in the social context which is using the IT, and therefore may not be artifactual in our definition.)

Together with IT, there are many different artifacts that people create which reflect their organizational structures. Table I lists some examples of these artifacts. These artifacts reflect, or encode the organizational structure, but should not be construed to *be* the organizational structure. These artifacts represent the different rules and protocols by which the members of the organization may choose to behave. The accuracy of these representations is, of course, variable. These rules and protocols have been likened to grammars in languages (Wand and Weber 1995). The grammar metaphor is useful as an analogy because linguistic grammars vary among communities, undergo change, naturally conflict among versions and may be accurately or inaccurately represented by grammatical texts. These organizational artifacts may vary among the organization's communities, undergo change, naturally conflict among themselves and may be accurately or inaccurately represented by structural artifacts.

[1]To a specific degree, these concepts are based on the roots of the terms, the original Latin and Greek ideas. Organization comes from the same Greek and Latin root as organism and organic (an individual life form) but most commonly meant "tool" or "device", and in this sense was also applied to "living" body organs. Our sense of the term is a natural, living device with a purpose. Structure arises from the Latin word *structure*, meaning something put together, taken from the verb *struere*, to put together. Structure implies the act of organizing, the assembling of the people, and the connections made between the individuals in the organization.

Table I. Examples of artifacts that people create which reflect their organizational structures.

1 *Organization charts*, which graphically depict organizational members and their different kinds of relationships
2 *Personnel policies*, which define reporting lines (who is whose boss), job descriptions, rewards structures and payroll policies
3 *Union agreements*, which reflect the relationships between union members and others in the organization
4 *Standard Operating Procedures (SOPs)*, which are typically detailed functional policies that define important coordinated actions in the organization.
5 *Resource access policies*, such as travel justification, modes of travel (private jet or tourist-class), company cars, cellular phones, etc.
6 *Workspace division*, which includes the size of an individual or group workarea and the collocation of organizational members (whose office is next to whose).
7 Workarea attributes and resources, such as decoration and furniture quality, quantity or size; privacy (e.g., corner-office or open-plan cubical); and dining facilities.
8 *Information Systems* which determine access to information resources such as computer accounts, LAN membership, automatic channelling of inputs and outputs, and screen and paper form designs.
9 *Methods* for developing information systems, which determine who sets the goals, who participates in the design, what issues are considered and how the system elements are represented.

2.3 Conflicting Versions of Organizational Structure

The different structural artifacts overlap, and will sometimes encode the organization's structure in conflicting ways. For example, the payroll policies, personnel policies and organization chart (Table I) may encode an individual without influence in organizational strategy; yet the workspace division, attributes and information system may encode a structure in which that same individual bears essential responsibility for shaping organizational strategy. Neither artifact *is* the organizational structure, that structure is defined by the relationships between the individuals, not by any particular set of organizational artifacts. When the artifacts suggest conflicting versions of the structures, the reflection of the real structure is blurred and the determination of its shape is made more difficult.

The organizational artifacts may tell different stories about the organizational structures. Conflicts between these stories seem to arise most often when the organizational structure differs in reality from an "official" version of the organization. These "official" organizational artifacts reflect the version of the organizational structure story as told by a particularly privileged class of organizational members: usually relatively senior management. Similarly to the priest-class in a theocracy, this class of individuals is widely accepted to own the authority to determine organizational structure. Accordingly, this class controls a large set of overt organizational artifacts.

However, when the structural story suggested by these "official" artifacts conflicts with the structural story suggested by many other organizational artifacts, then the reality of organizational structure is indistinct. Artifacts that may be beyond the influence of the "priesthood" in management may include those that are too menial for official control, such as workspace collocation; those that can be replaced by alternative artifacts (such as a lower-authority policy taking effect even though it contradicts and countermands a higher-authority policy); or those in which important functional artifacts ignore the specification artifacts, such as when the real information flows violate operating policies.

2.4 Conflicting Realities of Organizational Structure

An important aspect of conflicting artifactual reflections of organizational structure is the broad acceptance of "official" artifacts even when these conflict blatantly with the majority of other artifactual representations. This aspect regards the important, almost priestly power of the privileged organizational classes to interpret and pronounce organizational reality. As a result, management itself and many organizational scientists will not look beyond such privileged organizational artifacts, and sanction organizational structure to be, in reality, as officially declared by a certain subset of organizational artifacts.

The process by which an imaginary belief becomes accepted as being real, is the well-known thesis of Berger and Luckmann (1967) in the social construction of reality. Their thesis, applied here, suggests that the structural reality of organizations arises from routine relationships that become habitualized and explained in a symbolic universe. This symbolic universe in organizations may be strongly influenced by a powerful set of managers (the priestly class), and which can justify the relationships symbolically even when these become unnecessary or harmful (*e.g.*, government clerks following patently absurd bureaucratic rules because these justify some other part of the bureaucracy).

2.5 Emergent organizations

Organizational artifacts may also temporally reflect multiple versions of the reality of organizational structures. Each of these versions must be seen as being dynamic to a certain degree. That is, the relationships between organizational members (being social) are continuously changing. Emergent organizations are always seeking, but never quite achieving a regular pattern of behavior (Truex and Klein 1991). The most recent changes are less likely to be reflected in the organizational artifacts, and this means that such artifacts will be more-or-less out of sync with the reality of organizational structure. If the pace of change is fast in an organization, usually in response to fast-paced changes in its environment, then one should expect less fidelity and more conflict in its organizational artifacts with regard to its structure.

From this perspective, the issue runs deeper than merely conflicting versions of the reality of organizational structures, or indeed multiple organizational realities, and develops the possibility that no matter which artifacts one chooses to believe, that reflection of organizational structure is inevitably out of sync with the realities. This suggests that all organizational artifacts should be viewed with a degree of suspicion, especially in a setting with rapid change (a frequent characteristic in ISD). Each artifact tells a particular version of a particular set of organizational structures at a particular time. Indeed, such artifacts may have been entirely invented to match a desired organizational structure that may, or may not, have been realized at a later time. The relationship between these artifacts and current organizational structures is always open to question. As a consequence, contingency theories (*e.g.*, Davis 1982) provide an overly confining framework for ISD (Baskerville, Travis and Truex 1992).

3. THE CHALLENGE TO METHOD ENGINEERING

The problems of conflict between structural artifacts and emergent organizations is found at two levels of abstraction within the IS literature. At the first level, the IS community must deal with the potential conflict between the information technology, itself a structural artifact, and the emergent organization. At this fundamental level, one discovers IT that is ineffective in various ways, outdated, misplaced, or altogether unused because it conflicts with the present organizational structure. This conflict is dealt with, although somewhat indirectly, by the literature on IS failure, (*e.g.*, Bostrom and Heinen 1977, Ginzberg 1981, Lyytinen 1988 or Lyytinen and Hirschheim 1987), end-user development (*e.g.*, McLean 1979, Sumner and Kleer 1987, Galletta and Heckman 1990, or Amoroso and Cheney 1992) and software maintenance (*e.g.*, Schneidewind 1987 or Schnebeger 1995).

At the second level of abstraction, the IS community must deal with the potential conflict between the emergent organization and the development method (also itself a structural artifact) used to determine the structural artifact of information technology. At this level, one also discovers the IT development approaches are ineffective similarly to the IT itself, outdated, misplaced or altogether unused. This conflict is dealt with, also a bit indirectly, as a problem in need of solution by contingency approaches (*e.g.*, Davis 1982), prototyping approaches (*e.g.*, Naumann and Jenkins 1982), participative approaches (*e.g.*, Kyng 1991), and object-oriented approaches (*e.g.*, Coad and Yourdon, 1991). This conflict has been more directly dealt with in research which has shown that methods may not be entirely succeeding as a paradigm for the development of information technology (*e.g.*, Baskerville, Travis and Truex 1992, Wynekoop and Russo 1993, or Naur 1993).

Method engineering introduces a third level of abstraction, a method for creating methods. Indeed, method engineering may be a reaction to the structural conflicts which have (perhaps inevitably) accompanied the first two levels. The inability to discover suitable structural artifacts at the first level leads the search for suitable structures at the second level. That is, consistent failures at structuring IT as a match to organizations has demanded a search for successful structures for the structuring process. This idea is related to Giddens' (1984) social structuration theory, and has been explored directly in the IT context (Orlikowski and Robey 1991). Method engineering raises the problems of conflicts between structural artifacts and organizations to a higher level. See Figure 1.

However, it may be possible that this third level of abstraction will enable IT researchers to consider ever more essential structural artifacts regarding IT development and IT systems. If the methodical (the predefined, repetitive process) is abandoned to the second level, what structural artifacts remain for method engineering? Among the most prominent are notation and criteria.

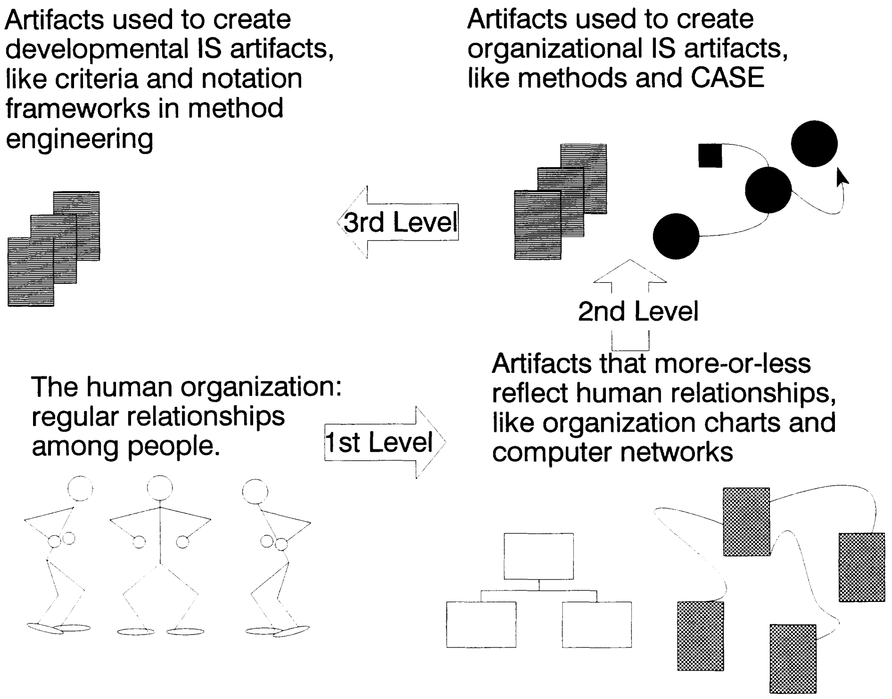
Artifacts used to create developmental IS artifacts, like criteria and notation frameworks in method engineering
Artifacts used to create organizational IS artifacts, like methods and CASE
3rd Level
2nd Level
Artifacts that more-or-less reflect human relationships, like organization charts and computer networks
The human organization: regular relationships among people.
1st Level

Figure 1. Three levels of information systems abstraction above the regular relationships in human organization.

The notation used in ISD methods varies widely: examples include data flow diagrams, data dictionaries, rich pictures, root definitions, and entity-relationship diagrams. The selection of the set of notation to be used in a system development project is among the most critical, since this decision will determine what concepts can (and what concepts cannot) be represented in the formal specification and design (Gause and Weinberg 1989). Conflicting representational schemes have also been explored in comparative methods studies, such as the work arising from the CRIS working conferences using the so-called IFIP case as its benchmark (Olle *et al.* 1988). Indeed, the empirical work by Bansler and Bødker (1993) suggests that the notation may be the only durable component of a systems development method.

The criteria regards the underlying rationale of the method, the details of its major aims, purposes and scope. The authors of the various methods are usually quite clear in explaining their criteria, but only in the context of each element of their method. As a whole, these criteria indicate what the method's authors believed would characterize a successful ISD project. But like philosophical assumptions, the discovery and general classification of these abstract criteria are problematic because these are conflated with the features of the method (*cf.* Coad and Yourdon, 1991 with Checkland and Scholes, 1990). However, there are examples of comparative work that has focussed on the features of methods (*e.g.* Olle, *et al.* 1983), and also work which has considered the underlying criteria (*e.g.* Olle *et al.* 1982).

At the third level of abstraction, namely method engineering, it will be necessary to introduce new structural artifacts. These new artifacts are likely include elements for selection of notation and criteria at the second level (ISD methods). Comparative studies of both notation and criteria will be prominent in formulating these new, third-level structural artifacts. However, the analysis should move beyond the description of the present structural artifacts of ISD, and also consider the real human organization and the way ISD unfolds in these organizations. Clearly, studying "methods" is not adequate for studying the reality of "ISD" (*cf.* Parnas and Clements, 1986, Baskerville, Travis, & Truex, 1992, Bansler & Bødker, 1993, Naur, 1993, Wynekoop & Russo, 1993). In other words, before building new artifacts that reflect the old ones on a new level of abstraction, one should question the old artifacts. In what ways do the structural artifacts of ISD conflict with the real organization?

There are a number of ways to prescriptively approach this issue. At one level, one can ask what criteria and notation might be available to capture conflicts between organizational structural artifacts (*e.g.,* when customer service policies and trouble-call operational procedures disagree), or to capture conflicts organizational structural artifacts and the organization itself (*e.g.,* when customer service policies and the actual social behavior of the service representatives disagree).

At another level (the method engineering level), one may ask what criteria and notation might be available to capture ever-present change in requirements. These elements would regard the need to follow an emergent organization through the course of an ISD project, especially considering the limited accuracy of all structural artifacts that purport to represent the organization in such settings.

4. NOTATION AND CRITERIA EXEMPLAR: SYSTEMS SECURITY

Systems security is an example of the problematic issues that arise from these challenges. Despite widespread agreement about the importance of privacy, reliability and integrity in information systems, explicit security constructs are extremely rare in ISD methods (Baskerville 1993a). Perhaps this is not so surprising considering that security features will typically conflict with the functionality of the system (Baskerville 1992). Also, security features are designed to prevent unpredicted organizational behavior, and will typically embody an uncomfortable constraint on emergent organizations (Baskerville 1993b). Finally, security features have been shown to create recursively security problems on their own (Baskerville 1995).

Not surprisingly, there are many examples of security problems in information systems (Neumann 1995). Perhaps it is more surprising that, despite the lack of consideration by ISD methods, the majority of information systems have security safeguards in place (albeit minimum), such as data backup and simple password schemes (*cf.* Hitchings 1995, Wood 1995). While this does not mean that existing security is entirely adequate for most information systems, it does mean that security features are being constructed into systems despite the lack of explicit structures in most ISD methods.

There are two implications that arise in this commonplace design of security features outside of explicit method artifacts. First, this design activity suggests that the structures of current ISD methods do not represent the reality of this aspect of ISD design. Second, this design activity might be better enabled if the structures of the ISD methods agreed more explicitly with the behavior of ISD analysts and designers.

These two implications comprise the security imperative for method engineering. This imperative regards the demand for explicit structures for developing system security in most ISD methods. On the one hand, this demand is founded on the unstructured design activity that is ignored by the criteria and the notation within IS design. On the other hand, this demand is founded on the need to reduce the substantial damage encountered by many organizations due to the lack of adequate security safeguards properly designed into their information systems.

The security imperative would entail, at a minimum, the inclusion of explicit security criteria and notation at the third level of abstraction. That is, method engineering should have structural artifacts that prescribe various criteria for security design and development, and alternative sets of notation for explicitly capturing security risks and protective features in the system.

These criteria and notation do not to be entirely invented. A survey of various security analysis and design notation is described by Baskerville (1993a), and book-length security methods exist (*e.g.*, Fisher 1984, Lane 1985). There are also surveys of criteria (*e.g.* Neugent 1982), and the European initiative on criteria for security of information technology is a "harmonization" of several national policies on such criteria (Commission of European Communities 1990). However, these resources provide only contrasting descriptions of second-order notation and criteria. These must be further analyzed in order to draw out frameworks for comparing and choosing among competing notations and criteria.

4.1 Notation

As a basis for one third-level notation framework, for example, the following security literature regarding methods for security safeguards specification was reviewed and analyzed using a limited inductive classification approach (*cf.* Sandman Klompus and Yarrison (1985). The framework below provides one initial possible framework for structuring the selection of security notation. The publications underlying this limited analysis are Browne (1979), Krauss (1980), Fisher (1984), Lane (1985), Baskerville (1988), Hutt *et al.* (1988), Fitzgerald and Fitzgerald (1990), Farquhar (1991), Ozier (1992), Forcht (1994) and Neumann (1995). Because of the heterogeneous nature of this body of literature, many other competing frameworks are possible. However, this example demonstrates the feasibility of such frameworks. The framework consists of three types of notation that comprise security features in ISD: Representation of security elements, security analysis and design, and security maintenance and management. This framework is illustrated in Table II.

This framework of security notation exemplifies a third-level structural artifact for organizational IS. Other related artifacts in method engineering might include heuristics for selecting a complete notational set, perhaps matching characteristics of the IS setting (captured in still another artifactual notation), or against the security criteria.

Table II. Third-level security notation framework for method engineering.

Security element representation notation.
The three elements organized for analysis in security design methods are typically the risks that threaten the system, the system assets requiring protection and the potential safeguards that might be erected to protect the system assets. Representation tools must be capable of representing inventories of these elements. Each of these inventories will need a classification system and a notation framework to permit analysis if they are to be practical. Examples of these classes and frameworks:

Risk		
	Categories:	natural disasters, malfunctions, criminal acts, errors
	Framework:	probability, cost of damage, degrees of impact
Assets		
	Categories:	computers, communications, storage, personnel
	Framework:	location, value, visibility, accessibility
Safeguard		
	Categories:	operating integrity, backup, access control, error detection
	Framework:	implementation details, costs, second-order problems

Security Analysis and Design Notation
The notation must also capture the rationale leading to the implementation of safeguards. For the purposes of both designing and maintaining the safeguards, the notation should capture the risks against asset mapping process, and the safeguards selection process in the sense that one should be able to implosively and explosively audit the design. For example, one should be able to trace each safeguard to the risks it protects against and the assets it is protecting. Likewise risks and assets should be mapped to the safeguards, or alternatively the notations should demonstrate that the benefit of safeguards were trivial and unnecessary. Examples of such notation:

- Risk ranking notation
- Asset ranking notation
- Risk-asset-safeguard mapping
- Safeguard ranking notation
- Safeguard selection and design integration details

Security Maintenance and Management Notation
Under the assumption that organizations are emergent, and that system security is yet another form of structural artifact in organizations, safeguards maintenance and maintenance of security may imply additional necessary notation. Examples of such notation:

- Security maintenance frameworks
  - Risks review and update process and routine
  - Safeguards validation process and routine
  - System maintenance security validation reviews

- Disaster planning frameworks
  - Computer center loss plan
  - Communication loss plan
  - Cross-training plan

4.2 Criteria

The security criteria is another third-level structural artifact that represents the security characteristics which method engineering might instill in the ISD method. An example of such third-level criteria was developed in a workshop sponsored by the INFOSEC Standards and Initiatives group of the Communications Security Establishment for the purposes of commenting on a federal guideline for risk management of computer information systems in the Government of Canada. The working group that regarded managerial considerations was particularly concerned with adopting a risk management method that was equally relevant in large, complex installations (*e.g.*, an air base), as well as small, simple installations (*e.g.*, a small, remote police post).

Rather than seek a single method, dictated from the central government security establishment, the working group concentrated on distributed security decision-making. That is, instead of completely defining the risk assessment process to be universally applied throughout the Government of Canada, the exact decision would be deferred to the localized agencies responsible for the computing elements. This deferral takes on some characteristics of third-level method engineering. The summary of the recommendations appear in Verrett and Hysert (1993).

The approach recommended by the working group was oriented toward centralized criteria-setting, rather than centralized control. The central authority would set the criteria for the process of risk assessment, and defer the determination of the exact method to the local agency. Even the specification of a range of techniques together with criteria for choosing among these techniques was seen to exclude the use of ideal, unique approaches that might occur to managers "on the scene". Instead, the recommendations suggested the criteria by which a highly qualified manager in the field might determine whether the risk management process was successful. While examples of risk assessment techniques were suggested, it was not mandatory to choose one of the examples. The local manager would be free to innovate in situations where such innovation, in the judgement of that local manager, seemed to be required. The criteria are described in Table III.

The criteria in Table III represent another third-level structural artifact, and illustrates one alternative set of criteria. A more complete set of security criteria at this method-engineering level might offer alternative sets of criteria, perhaps for different levels of security, or different kinds of organizations, (government, manufacturing, retail, financial, *etc.*). The chosen criteria might then be used as a measure of the performance of method engineering in determining the security features (*e.g.* notation) of the second-level method.

4.3 Linking The Imperative Through The Levels

The imperative leads to the addition of notation and criteria to the method. The imperative may take various forms or degrees. Contrast, for example, relatively stronger, highly structured and inflexible security in military situations with relatively weaker, less structured and more flexible security in consumer goods manufacturing situations. The third-level artifacts must support the construction of second-level artifacts that respond to both of these contrasting situations (as well as others). Third-level method engineering must be capable

Table III. Criteria for risk management (adapted from Verrett and Hysert, 1993).

The process should be goal-directed.
The security goals of the government agency must be established at the beginning of the project, must meet applicable policies and standards, must involve the system owner from the start, must define resource constraints up-front, and must be as non-obstructive as possible. These goals should be 'appropriate' for the government group as judged by the managers in charge. The system's nature might take into account the criticality of failures (for example, a police cruiser dispatch system is more critical than a campground firewood management system), or the environment of the system (broad public access or highly restricted access to the system). Examples of such goals include:

- "trouble free operations"
- "minimum events"
- "highly private"

There must be a reasonably exhaustive threat analysis.
The process of risk analysis must include a threat analysis that considers a 'reasonable' range of threats. For a highly critical system, such a 'reasonable' range would necessarily be more exhaustive than non-critical systems. This will typically mean that the process will involve the use of an analytic model (*e.g.*, the US National Institute of Standards and Technology Model) and a reasonable tool or guideline (such as the CCIT Risk Analysis Methodology, CRAMM, a software-based approach).

The process must be updatable and reusable.
The process of risk management must result in the establishment of a permanent maintenance cycle for system security. This typically means that routine security reviews must be held which consider changes in security needs as a result of continued operation of the system. In many situations a security maintenance review plan will be needed that produces a periodic report certifying that the security is still intact. The cognizant manager must know what action is necessary if the report fails to materialize, or appears inadequate.

The process must achieve closure in both certification and accreditation.
The risk management process must conclude with an event of some type that embodies the instance at which the initial risk management project is successfully concluded. That is, such a risk management process must lead to a certifiable result.

The results must be repeatable.
The risk management process must be "logically" repeatable. This does not necessarily mean that anyone actually expects to repeat the process and compare results. Rather, it means that the process must be thoroughly understood as it occurred, and it must be carefully documented. In this way the decision-making process can be reconstructed such that higher management might be able to review whether the decisions about risks were made in a reasonable fashion in a given situation.

of constructing the second-level artifacts (the method) such that they are capable of producing first-level artifacts (information systems) which respond to the human organization. The nature of that response depends of the particular view of the relationship between the information system as a structural artifact and the human organizational structure. Five such views are discussed in the appendix. This illustrates how the ideal set of structural artifacts of method engineering would reflect the intersection of several dimensions: the set of possible imperatives, the forms and degrees of those imperatives, an open set of possible second-level structural artifacts, the set of viewpoints on the relationship between IS artifacts and the organization, and the strength of the harmony among the organization's structural artifacts and between those artifacts and the human organization.

This multi-dimensional nature of third-level structural artifacts introduces some complexity, but this certainly is not imponderable. As an example of how this analysis can be applied to the security imperative, assume the viewpoint (from the appendix) that IT can be used to shape the organization, and strong security is the imperative. Under these conditions, then the ISD method should be characterized by notation and criteria that will lead to prescriptive structural artifacts. Such prescriptive security artifacts within the IS include enforced access controls, which in turn require that the method's notation be characterized by extensiveness in the details in its safeguard categories, safeguard frameworks, design integration and validation routines. In addition, the third-level criteria suggests the need for a detailed goal set, more rigorous forms of threat analysis, such as one that includes automated threat databases (*e.g.,* CRAMM). The criteria under this imperative also indicate artifacts like a formal security maintenance review cycle, initial certification and routine recertification, and closely detailed documentation about the design decisions. Further, the assumed organizational viewpoint pins the achievement of strong security on a fair amount of harmony between the organization's structural artifacts and the structure of the human organization.

As a second, contrasting example of how this analysis can be applied to the security imperative, assume the emancipatory viewpoint (from the appendix), that IT involves ethical decisions about structuring the workplace. This example will also assume the contrasting position that security is fairly weak imperative. Under these conditions, then the ISD method should be characterized by notation and criteria that will lead to the fewest and least constraining, perhaps only implicit, structural artifacts. Such implicit security artifacts within the IS might only include training, simple passwords and data backup routines. These in turn imply that the method's notation be loosely structured, perhaps only free text in a few brief, suggested sections (*e.g,* risk, asset and threat rankings). In addition, the third-level criteria suggests only a few broad security goals will be involved and an informal, participative threat analysis. The certification and routine reviews will probably be internal, informal, and participative. Further, the assumed organizational viewpoint pins the achievement of reasonable security on a fair amount of highly-motivated participation within the organization itself, and that the structural artifacts related to security must be ethical with regard to the workplace.

5. SUMMARY AND FUTURE RESEARCH

The above discussion demonstrates how method engineering introduces a new level of structural artifact into human organizations. By selecting security as the example for the demonstration, this discussion also highlights the potential for method engineering not only to "situationalize" methods, but to correct general oversights in many of the existing published methods. That is, structural artifacts at the third level may respond to a general set of system imperatives which must be adapted to the development situation. This general set of system imperatives may be more complete, and therefore lead to more complete ISD.

Method engineering represents a third level of abstraction in ISD. This higher level of abstraction increases the need to understand the relationship between human organizations, organizational structure, and structural artifacts. Structural artifacts include IT, ISD methods and method engineering. The resolution of conflict between such artifacts and the organization, and between the artifacts themselves, motivates the introduction of the third-level artifacts of method engineering.

Further research is needed to understand the degree to which method engineering artifacts might conflict with organizational structure, or with each other. The human organizations in question not only include the target organization for the IT design, but also the organizations involved in ISD and method engineering itself. Also the impact of this conflict on the success of method engineering remains an open question until the application of method engineering grows. Additional work is also needed to determine what other imperatives should comprise the structural artifacts of method engineering. Examples might include usability, availability, timeliness, *etc.* In addition, the security exemplar also reveals the need for further research to determine the broad set of security criteria, notation, and other third-level structural artifacts necessary to implement the security imperative in method engineering.

APPENDIX A: VIEWPOINTS OF IT AND ORGANIZATION STRUCTURE

At least five separate viewpoints of the relationship between IT and organizational structure can be distinguished in the literature. Admittedly, each of these viewpoints is somewhat abstract, like stereotypes or caricatures. These will not be found in pure form "in the wild" of real organizational management. However, these theoretic viewpoints inhabit, and may even dominate other theories, practical trends and models of IS and IT. The discussion of each viewpoint will consider its central characteristic, the role of IT and IS under this viewpoint and an example of recently published research that relates critically to this viewpoint.

A.1 IT As A Medium for Organizational Communication

This viewpoint is characterized by the idea that information is a commodity in a similar sense to electricity, water and gas. The function of IT and IS is similar to that of an information utility providing an economical and sufficient supply of good quality information necessary to the organization. The IS parts of the organization act similarly to a sort of the utility company that sets up the data repositories and flow lines as needed in the organization.

As an example of where this viewpoint currently holds a strong influence, consider outsourcing. Practical and research publications on outsourcing often presume that the information utility can be contracted out, like the telephone switchboard and housekeeping. For example, Willcocks and Fitzgerald describe common problems discovered by organizations in their attempts to contract out, in varying degrees, their information technological support (Willcocks and Fitzgerald 1994).

A.2 Strategic Use of IT

This viewpoint is characterized by the assumption that information can be a central organizational product, or an essential enabling factor in a central organizational product. The role of IT and IS is tightly connected with the goals, strategies and purpose of the organization itself. Under this viewpoint, the organization could not produce its products competitively without IT.

The primary examples of this viewpoint include the American Airlines Sabre and American Hospital Supply case studies. These are now also iconic representations of cases where a few success stories dominate a management trend, followed by suspicions that a large number of attempted emulations resulted in failure. Publication readership seems intensely interested in the innovative successes and not in the emulation failures. For example, Kettinger *et al.* (1991) survey the long-range impact of a number of strategic IT systems. For a further example of this viewpoint, see Reich and Benbasat's (1990) study of customer-oriented strategic IT.

A.3 IT As A Mechanism for Shaping Organizations

Other parts of the IT and IS literature are characterized by the instrumental idea that the organization itself can be restructured by restructuring its IT. That is, the shape of the organization will follow the form of its IT. If one reorganizes the IT, one thereby reorganizes the human behavior. The role of IT and IS is therefore one of directing the organizational resources, both enabling and constraining the organizations purposeful work to the paths determined by management.

This view is typified by some of the current writings in Business Process Reengineering. The organization is moved into a new form by destroying the old IT (Hammer 1990), which embodied the old economic-specialization Taylorism, and rationally building a new process-centric organization that is effectively enabled by advanced IT (Davenport and Short 1990). Here again one encounters a similar problem to that of strategic IS, in that the literature is dominated by fairly limited set of success stories, while the practical community seems to be encountering serious problems emulating these successes (Manganelli and Klein 1994).

A.4 Matching IT To The Organization

This viewpoint is characterized by the assumption that the organizational structure is determined independently of its IT, and that successful IT will be shaped to match and support the structure of the organization. The role of IT is that of a tool that makes organizational processes easier under its preexisting structural constraints. The organizational processes must occur with or without IT, and if the IT does not help these processes, then the IT will be irrelevant, become ignored and fall into disuse. Successful IS is determined by its ability to shape itself to the needs of the organization.

This viewpoint is typical of the traditional IS development literature, with its focus on requirements elicitation and specification. Such literature will typically argue that lengthy systems analysis and data modelling is justified by the smooth conversion and enhanced lifespan of the new system (*cf.* Lyytinen 1987).

A.5 IT For Emancipation

This viewpoint is characterized by its focus on the human and social implications of the use of IT. It strongly shaped the socio-technical literature in systems development, with its recognition that IT choices carried ethical determinations in structuring the human workplace. IT could make worker's lives better, worse, or unnecessary. IS design and management was both a social and a technical act.

This viewpoint is typical of the socio-technical literature in IS development, and the trade-union influence in North European IS research. These assumptions dominated some systems development methods, like ETHICS (Mumford 1983) and cooperative prototyping (Er 1987), and is currently found in some of the work in systems development that focusses on the worklife of the developer (*cf.* Hirschheim and Klein 1994).

6. REFERENCES

Amoroso, D. and P. Cheney (1992) Quality end user developed applications: some essential ingredients, *Database 23* (1) (Winter) 1-12.

Avison, D. and G. Fitzgerald (1988) *Information Systems Development: Methodologies, Techniques and Tools.* Oxford: Blackwell Scientific.

Bansler, J. and K. Bødker (1993) A reappraisal of structured analysis: Design in an organizational context, *ACM Transactions on Information Systems 11* (2) 165-193.

Baskerville, R. (1988) *Designing Information Systems Security.* Chichester: Wiley.

Baskerville, R. (1992) The developmental duality of information systems security, *Journal of Management Systems 4* (1) 1-12.

Baskerville, R. (1993a) Information systems security design methods: Implications for information systems development, *Computing Surveys 25*, (4) December 375-414.

Baskerville, R. (1993b) Information systems security: Adapting to survive, *Information Systems Security 2* (1), 1993, 40-47. Reprinted, as New approaches to information systems security in Umbaugh, Robert (Ed.) *Handbook of IS Management 1994-95 Yearbook.* New York: Auerbach, 1994, pp S257-S265.

Baskerville, R. (1995) The second order security dilemma, in Orlikowski, W., Walsham, G., Jones, M., and DeGross, J. (Eds.) *Information Technology and Changes in Organizational Work*. London: Chapman & Hall, pp. 239-249.

Baskerville, R., J. Travis, and D. Truex (1992) Systems without method in Kendall, K. Lyytinen, K. and DeGross, J. (Eds.) *IFIP Transactions on The Impact of Computer Supported Technologies on Information Systems Development*. Amsterdam: North-Holland, pp. 241-270.

Berger, P. and T. Luckmann (1967) *The Social Construction of Reality, A Treatise in the Sociology of Knowledge*, Penguin Books.

Bostrom, R. and S. Heinen (1977) MIS problems and failures: A socio-technical perspective, Part I: The causes, *MIS Quarterly*, (September), 17-32, and MIS problems and failures: A socio-technical perspective, Part II: The application of socio-technical theory, *MIS Quarterly*, (December 1977), 11-28.

Browne, P. (1979) *Security: Checklist For Computer Center Self-Audits*. AFIPS, Arlington, Va.

Checkland, P. and J. Scholes (1990) *Soft Systems Methodology in Practice*. Chichester: J. Wiley.

Coad, P. and E. Yourdon (1991) *Object-Oriented Analysis 2nd Ed.*. Englewood Cliffs: Yourdon.

Commission of European Communities (1990) *Information Technology Security Evaluation Criteria (ITSEC), Provisional Harmonized Criteria, Version 1.2*. Brussels, Belgium: Commission of European Communities, Directorate--General XIII (June).

Connell, J. and L. Shafer (1989) *Structured Rapid Prototyping: An Evolutionary Approach to Software Development*. Englewood Cliffs: Yourdon Press.

Davenport, Thomas and James Short (1990) The new industrial engineering: Information technology and business process redesign, *Sloan Management Review* (Summer) 11-27.

Davis, G. (1982) "Strategies for information requirements determination," *IBM Systems Journal 21* (1) 4-30.

Embley, D., B. Kurtz and S. Woodfield (1992) *Object-Oriented Systems Analysis: A Model-Driven Approach*. Englewood Cliffs, N.J.: Yourdon Press.

Er, M. (1987) Prototyping, participative and phenomenological approaches to information systems development, *Journal of Systems Management* (August) 12-15.

Farquhar, B. (1991) One approach to risk assessment, *Computers & Security 10*, 1, 21-23.

Finkelstein, C. (1989) *An Introduction to Information Engineering: From Strategic Planning to Information Systems*. Sydney: Addison-Wesley.

Fisher, R. (1984) *Information Systems Security*. Englewood Cliffs: Prentice-Hall.

Fitzgerald, J. and A. F. Fitzgerald (1990) *Designing Controls Into Computerized Systems*. Jerry Fitzgerald & Associates, Redwood City, Ca.

Forcht, K.A. (1994) *Computer Security Management*, Danvers, Massachusetts: Boyd & Fraser.

Galletta, D. and R. Heckman (1990) A role theory perspective on end-user development, *Information Systems Research 1*, (2) (June) 168-187.

Gause, D. and G. Weinberg (1989) *Exploring Requirements: Quality Before Design* New York: Dorset House.

Giddens, A. (1984) *The Constitution of Society: Outline of the Theory of Structure*. Berkeley, Calif: Univ. of California Press.

Ginzberg, M. J. (1981) Early Diagnosis of MIS Implementation Failure: Promising Results and Unanswered Questions, *Management Science 27*, (4).

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

Hirschheim, R. and H. K. Klein (1992) Paradigmatic influences on information systems development methodologies: Evolution and conceptual advances. *Advances in Computers 34*, 294-381.

Hirschheim, R. and H. K. Klein, (1994) Realizing emancipatory principles in information systems development: The case for ETHICS, *MIS Quarterly 18* (March) 83-95.

Hitchings, J. (1995) Deficiencies of the traditional approach to information security and the requirements for a new methodology. *Computers & Security 14* (5), 377-383.

Hutt, A. E., S. Bosworth and D. B. Hoyt (eds.) (1988) *Computer Security Handbook*. Macmillan Publishing Co., New York, NY.

Jackson, M. C. and P. Keys, (1984) Towards a system of systems methodologies. *Journal of The Operational Research Society 35*, 473-486.

Jayaratna, N. (1988) Guide to methodology understanding in information systems practice. *International Journal of Information Management 8*, 43-53.

Jayaratna, N. (1993) Methodology assistance in practice: A critical evaluation. *Systemist 15*, (1) February, 5-16.

Kettinger, W., V. Grover, S. Guha, and A. Segars (1994) Strategic information systems revisited: A study in sustainability and performance. *MIS Quarterly 18* (1) (March) 31-58.

Krauss, L. I. (1980) *SAFE: Security Audit And Field Evaluation For Computer Facilities And Information*. AMACOM, New York, NY.

Kumar, K. and R. Welke (1992) Methodology engineering: A proposal for situation-specific methodology construction, in W. Cotterman, and J. Senn (Eds.) *Challenges and Strategies for Research in Systems Development*. New York: John Wiley & Sons, pp. 257-268.

Kyng, M. (1991) Designing for cooperation: Cooperating in design, *Communications of the ACM 34* (12) (December) 65-73.

Lane, V.P. (1985) *Security of Computer Based Information Systems*. London: Macmillan.

Lyytinen, K. (1987) Different perspectives on information systems: Problems and solutions, *ACM Computing Surveys* (1) (March) 5-42.

Lyytinen, K. (1988) Expectation failure concept and systems analysts view of information system failures: Results of an exploratory study, *Information & Management 14*, 45-56.

Lyytinen, K. and R. Hirschheim (1987) Information systems failures: A survey and classification of the empirical literature, *Oxford Surveys in Information Technology 4*.

Manganelli, R. and M. Klein (1994) Should you start from scratch? *Management Review 83* (7) (Jul) 45-47.

McLean, E. R. (1979) End users as application developers, *MIS Quarterly 3* (4) (December) 37-46.

Mumford, E. (1983) *Designing Human Systems For New Technology: The ETHICS Method*. Manchester: Manchester Business School.

Naumann, J. and A. Jenkins (1982) Prototyping: The new paradigm for systems development, *MIS Quarterly* (Sept) 29-44.

Naur, P. (1993) Understanding Turing's universal machine: Personal style in program description. *The Computer Journal 36* (4) 351-372.

Neugent, W. (1982) Acceptance criteria for computer security, *NCC Conference Proceedings.* Arlington, Va: AFIPS Press.

Neumann, Peter G. (1995) *Computer Related Risks.* New York: ACM Press.

Oliga, J. (1988) Methodological foundations of systems methodologies. *System Practice, 1* (1) (March), 87-112.

Olle, A., J. Hagelstein, I. Macdonald, C. Rolland, H. Sol, F. Van Assche, and A. Verrijn-Stuart (1988) *Information Systems Methodologies: A Framework for Understanding.* Wokingham: Addison Wesley.

Olle, T. W., H. G. Sol and A. A. Verrijn-Stuart, (1982) (eds) *Information Systems Design Methodologies: A Comparative Review*, Amsterdam: North Holland.

Olle, T. W., H. G. Sol and C. J. Tully, (1983) (eds), *Information Systems Design Methodologies: A Feature Analysis*, Proceedings of the IFIP WG 8.1 Working Conference on Feature Analysis of Information Systems Design Meeting, York, UK, 5-7 July, 1983, Amsterdam: North-Holland.

Orlikowski, W. and D. Robey (1991) Information technology and the structuring of organizations, *Information Systems Research 2* (2) (June) 143-169.

Ozier, W. (1992) Risk Assessment and Management *Data Security Management* Report 85-01-20. New York: Auerbach.

Parnas, D. and P. Clements (1986) A rational design process: How and why to fake it. *IEEE Transactions on Software Engineering SE 12* (2), February, 251-257.

Reich, B. and I. Benbasat (1990) An empirical investigation of factors influencing the success of customer-oriented strategic systems. *Information Systems Research 1* (3) (September) 325-347.

Sandman, P., C. Klompus and B. Yarrison (1985) *Scientific and Technical Writing.* Ft. Worth, Texas: Holt, Rhinehart and Winston.

Schnebeger, S. (1995) Distributed computer system complexity versus component simplicity. Its effects on software maintenance. Georgia State University Manuscript, summarized in J. DeGross, G. Ariav, C. Beath, R. Hoyer and C. Kemerer (eds.), *Proceedings of the Sixteenth International Conference on Information Systems.* New York: ACM Publ. p. 351.

Schneidewind, N. (1987) The state of software maintenance. *IEEE Transactions on Software Engineering SE-13* (3) March 303-310.

Sumner, M. and R. Kleer (1987) Information systems strategy and end-user application development, *Data Base 18* (4) (Summer) 19-30.

Truex, D. and H. K. Klein (1991) A rejection of structure as a basis for information systems development. In R. Stamper, R. Lee, P. Kerola and K. Lyytinen (Eds.), *Collaborative Work, Social Communications and Information Systems.* Amsterdam: North-Holland, pp. 213-236.

Verrett, R. and R. Hysert (1993) Summary of findings, working group 2, managerial and structural issues in the draft risk management framework. in *Proceedings 5th International*

Computer Security Risk Management Workshop. Ottawa: National Institute of Standards and Technology and Communications Security Establishment, 7-9.

Wand, Y., and Ron Weber (1995) On the deep structure of information systems, *Information Systems Journal 5* (3) (July) 203-223.

Willcocks, L. and G. Fitzgerald (1994) Toward the residual is organization? Research on it outsourcing experiences in the united kingdom. in Baskerville *et al.* (eds) *Transforming Organizations with Information Technology*. Amsterdam: North-Holland, pp. 129-152.

Wood, C. C. (1995) Identity token usage at American commercial banks. *Computer Fraud and Security Bulletin* (March) 14-16.

Wynekoop, J. and N. Russo (1993) System development methodologies: Unanswered questions and the research-practice gap, in J. Degross, R. Bostrom, and D. Robey (Eds.), *Proceedings of the 14th International Conference Information Systems*. New York: ACM Publ. pp. 181-190.

Yourdon, E. (1989) *Modern Structured Analysis*. Englewood Cliffs, NJ: Yourdon Press.

7 BIOGRAPHY

Richard Baskerville is an associate professor in the School of Management at Binghamton University. His research focusses on security and methods in information systems, their interaction with organizations and research methods. He is an associate editor of *MIS Quarterly* and *The Information Systems Journal*. Baskerville's practical and consulting experience includes advanced information system designs for the U.S. Defense and Energy Departments. He is vice chair of the IFIP Working Group 8.2, and a Chartered Engineer under the British Engineering Council. Baskerville holds MSc and PhD degrees from the London School of Economics.