Information Assurance for Enterprise Engineering

Jody Heaney, Duane Hybertson, Ann Reedy, Susan Chapin,
Terry Bollinger, Douglas Williams, and Malcolm Kirwan, Jr.
The MITRE Corporation
McLean, VA
heaney@mitre.org

Abstract

This paper describes the results to date of a research effort to apply a pattern approach to the problem of addressing information assurance (IA) in enterprise-level information engineering. IA is not effectively included in Enterprise Architectures today, largely because there is no compendium of knowledge immediately useful to enterprise engineers who are not IA specialists. The goal of this research project is to capture IA best practices from the policy level through implementation levels in a representation accessible to enterprise architects and engineers. We are using the successful paradigm of patterns to capture this best practice knowledge as well as the understanding of how and where it fits within the context of an enterprise architecture framework.

1. Problem

This paper describes the results to date of a research effort whose goal is to apply a pattern approach to the problem of integrating information assurance (IA)[1] into all levels of enterprise engineering. Motivations for the research include:

- The National Research Council identified IA as a national critical problem [NRC99] and the terrorist attack of 9/11 has compounded that problem.
- The rise of the Internet and e-commerce has increased both the visibility and the need for adequate IA at all levels of the enterprise

The industry is moving toward more formal development and documentation of Enterprise Architectures (EAs) based on Enterprise Architecture Frameworks. The purpose of a framework is to insure that all elements of the enterprise – including IA – are adequately addressed. This completeness is important since EAs are used as the basis for Information Technology (IT) management and investment decisions. One driver for EA use impacting many organizations is in the United States (U.S.) Government, where this use of EA is enforced by both legislation and regulations (Clinger-Cohen Act and Office of Management and Budget (OMB) Circular A-130, respectively).

[1] Information Assurance (IA) is information operations that protect and defend information and information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation [NSTISSI99]. The term *IA* has come into use in recent years to indicate security-related concerns that extend beyond the traditional scope of information systems security.

The purpose of an EA is to insure that IT effectively supports the needs of the business. It follows that the IT IA element in the enterprise architecture must be linked to the enterprise business needs.

IA must also be effective. Effective IA within an enterprise architecture depends on both integration and separate analyzability of IA elements. IA elements of the enterprise architecture must be linked to other elements of the architecture (integration), because IA elements do not exist in isolation. IA elements must also be viewable independently, because otherwise the effectiveness of the IA solution cannot be determined (separate analyzability).

Today IA is not being effectively addressed in most developed frameworks and EAs. What usually happens is one of two approaches. Frequently the IA aspects of a system are designed and analyzed separately from the rest of the architecture, and are therefore not well integrated, leading to a potential for new vulnerabilities. Alternatively, the IA aspects are designed implicitly into the architecture so that they cannot be extracted, and they are therefore not separately analyzable. The need to link IA aspects to business needs is often overlooked in both approaches.

Part of the problem is the lack of a compendium of IA knowledge immediately useful to enterprise engineers who are not IA specialists. Current practice segregates IA engineering from the remainder of engineering of a system or enterprise, such that IA engineers are obliged to address all aspects of IA. A shortage of qualified IA engineers means that IA issues cannot be properly addressed or integrated into decisions made at planning, policy, and business design levels of the enterprise. Thus, there is a need for practical guidance on how enterprise engineers not qualified as IA engineers can include IA in architectures. Developing such guidance should be possible because a significant portion of day-to-day IA work is routine and well understood by IA engineers, even though it is not understood by non-IA engineers.

Another part of the problem is that, unfortunately, available standards work does not support both integration and separate analyzability. Much of the early standards work for IA (i.e., security) was developed by and for the defense sector. Documents such as the Trusted Computer System Evaluation Criteria [DoD85], guidance for applying it [NCSC85], and the evolution to Federal Criteria [NIST-NSA92] are focused on system-centric approaches to system development, in which each system component is secured separately from the others. They are focused on environments in which all systems are assumed to be subject to one of a very few IA classes, determined by the classification of the data and authorizations of users that the systems handle. These standards do not clearly address other environments in which the needs for IA are unique to each modern business enterprise or extend beyond securing the information systems themselves. They do not provide for explicit links between the IA aspects of the enterprise architecture and the business needs aspects. In these standards IA is treated as separate from other architecture activities.

Today, security and privacy issues have gained greater focus even when the data are not classified, and public laws such as the Privacy Act [PL74] address a wider variety of Government systems. However, these statutes levy requirements but do not offer assistance with how the requirements may be satisfied.

In the commercial sector, guidance for incorporating IA into enterprise architectures has fared little better. Standards do exist that address areas such as the security of applications [NIST95] and audit [NIST94], offering concrete advice to security practitioners. However, these standards do not support integration of the IA aspects of the architecture with other aspects, and they do not link the concrete advice to the abstract needs as expressed in an enterprise architecture.

Consideration for a broader perspective began to appear in the last decade [NIST96, NIST01, CC99, Pipkin97, Cheswick94], but in these documents IA was still largely being treated as a separate element. It has recently become apparent that past attempts to achieve information security have too often failed, and the current situation of "penetrate and patch" is a very resource expensive conflict that cannot be won. Hence, some IA engineers have come to believe that the best hope for success is to go back and rethink information security in the context of enterprise architecture, and approach IA in a new, very fundamental way. This research is part of that movement, using the pattern paradigm to model the fundamental concepts of IA.

2. Research Project Objectives and Approach

2.1 Objectives

The purpose of this research project is to advance the state of the practice, not advance the state of the art; to "raise the floor" rather than "raise the ceiling." While others may be working on research to advance the state of the art in IA, we are attempting to capture IA best practices in use today and make them more accessible to the enterprise engineering and general systems engineering community.

More specifically, the objectives of the research project are to:

- Capture knowledge of routine, best practice solutions to standard IA problems across the full range of enterprise engineering levels of abstraction
- Integrate the IA solutions into a total IA view that spans all levels of abstraction
- Show how this IA view is integrated into the currently used enterprise views
- Make individual best practice IA solutions and the integrated IA view available in a representation accessible to engineers who are not IA specialists
- Provide guidance to engineering practitioners for applying the best practice IA solutions

2.2 Research Approach

The research team is capturing IA best practices decision support information in the form of patterns as a way of making it comprehensible and accessible to engineers who are not IA specialists. Patterns are an approach that has been successfully used in the systems and software communities to capture and share knowledge about well-known and successful solutions to common technical problems [Bushmann96; Gamma94]. The pattern format has also been extended to address organizations [Dikel00], configuration management [BrownW99], and analysis [Fowler96] problems, so patterns should be applicable to the wide range of abstraction levels needed for enterprise engineering.

The patterns we are developing will be organized into a system of patterns to provide the full IA view. The approach to the system of patterns that we are developing involves three innovative aspects. First, we are using a system of patterns to provide an IA perspective to an architecture framework. Second, we are extending the scope of current architecture and design patterns by integrating our IA patterns across multiple levels of abstraction from enterprise policy and business models to implementation options. Third, we are focusing the system of patterns more on decision support for selection among alternatives than on actual implementation, because IA decision information is one of the biggest deficits in current enterprise engineering.

Today, when EAs are being established, IA usually is not involved until well into development. Our research is intended to insure that IA is addressed across the full range of enterprise engineering levels by using the Zachman Framework as a guide. (See Figure 1.) The Zachman Framework is an established conceptual Enterprise Architecture Framework (more details are available in [Zachman87; Sowa92; ZIFA02]). It is the basis from which many of today's architecture frameworks are derived, and is a convenient context for organizing and integrating IA patterns. Discussion of the forces that motivated us to choose the Zachman Framework, and indication of possible alternatives, are provided in Section 2.3.

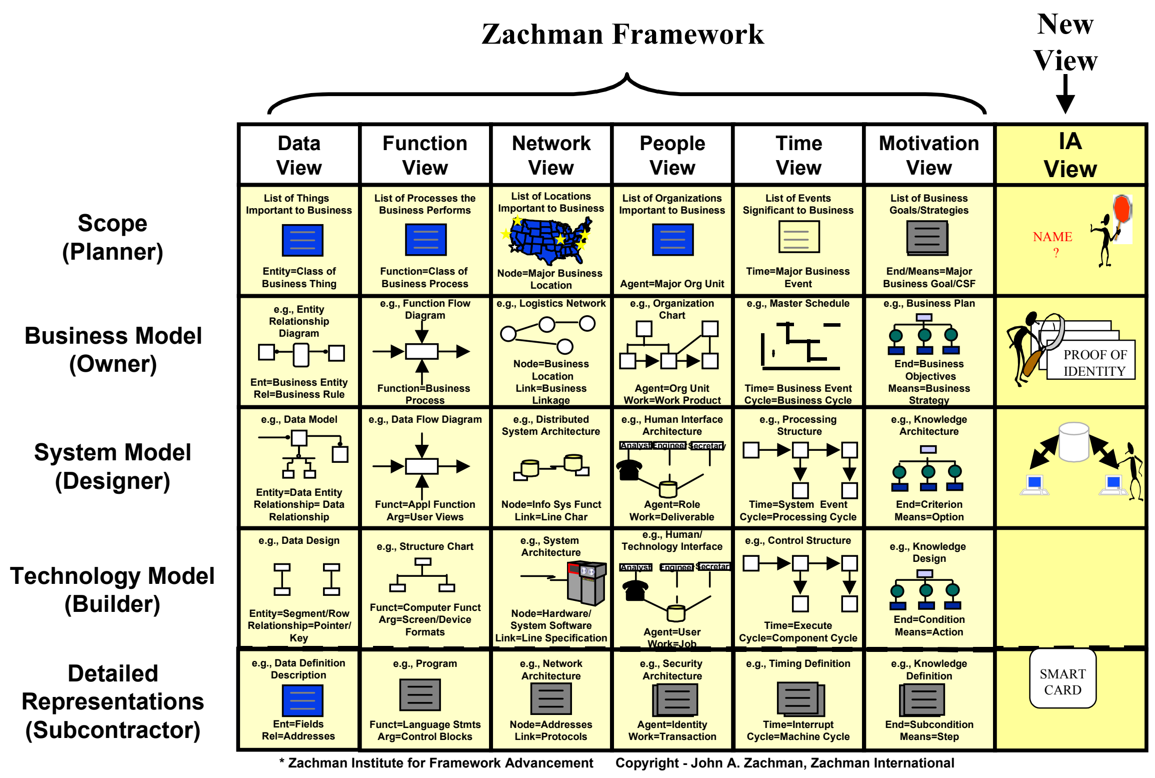
Zachman Framework
New View
Data View
Function View
Network View
People View
Time View
Motivation View
IA View
Scope (Planner)
Business Model (Owner)
System Model (Designer)
Technology Model (Builder)
Detailed Representations (Subcontractor)
List of Things Important to Business
Entity=Class of Business Thing
List of Processes the Business Performs
Function=Class of Business Process
List of Locations Important to Business
Node=Major Business Location
List of Organizations Important to Business
Agent=Major Org Unit
List of Events Significant to Business
Time=Major Business Event
List of Business Goals/Strategies
End/Means=Major Business Goal/CSF
NAME ?
e.g., Entity Relationship Diagram
Ent=Business Entity Rel=Business Rule
e.g., Function Flow Diagram
Function=Business Process
e.g., Logistics Network
Node=Business Location Link=Business Linkage
e.g., Organization Chart
Agent=Org Unit Work=Work Product
e.g., Master Schedule
Time= Business Event Cycle=Business Cycle
e.g., Business Plan
End=Business Objectives Means=Business Strategy
PROOF OF IDENTITY
e.g., Data Model
Entity=Data Entity Relationship= Data Relationship
e.g., Data Flow Diagram
Funct=Appl Function Arg=User Views
e.g., Distributed System Architecture
Node=Info Sys Funct Link=Line Char
e.g., Human Interface Architecture
Analyst Engineer Secretary
Agent=Role Work=Deliverable
e.g., Processing Structure
Time=System Event Cycle=Processing Cycle
e.g., Knowledge Architecture
End=Criterion Means=Option
e.g., Data Design
Entity=Segment/Row Relationship=Pointer/ Key
e.g., Structure Chart
Funct=Computer Funct Arg=Screen/Device Formats
e.g., System Architecture
Node=Hardware/ System Software Link=Line Specification
e.g., Human/ Technology Interface
Analyst Engineer Secretar
Agent=User Work=Job
e.g., Control Structure
Time=Execute Cycle=Component Cycle
e.g., Knowledge Design
End=Condition Means=Action
e.g., Data Definition Description
Ent=Fields Rel=Addresses
e.g., Program
Funct=Language Stmts Arg=Control Blocks
e.g., Network Architecture
Node=Addresses Link=Protocols
e.g., Security Architecture
Agent=Identity Work=Transaction
e.g., Timing Definition
Time=Interrupt Cycle=Machine Cycle
e.g., Knowledge Definition
End=Subcondition Means=Step
SMART CARD
* Zachman Institute for Framework Advancement
Copyright - John A. Zachman, Zachman International

Figure 1. Adding IA to the Zachman Framework

The Zachman Framework provides architectural views as vertical columns, and stakeholder information model perspectives as horizontal rows that correspond to the full enterprise system development cycle from planning through design to full implementation. Figure 1 shows the Zachman Framework enhanced with an IA view as an additional column. With this enhanced framework, one can consider an enterprise IA view in the same way one can consider an enterprise data view or functional view. The IA view addresses all levels of abstraction, from the enterprise scope to the enterprise technology model and detailed representations.

The IA view can be divided into "subviews" or "slices" each representing one area of IA. Since there is no consensus on an IA taxonomy, our initial taxonomy uses definitions provided in the earlier IA definition from [NSTISSI99] to support our system of patterns. Integration within the IA view is achieved via a system of patterns that captures the relations among IA elements across stakeholder perspectives. We are experimenting with using tree structures within each IA area to show these integrating relations and also to express the decision points in selecting among alternative solutions at each level.

The entire system of patterns resulting from our research will be organized into an IA Enterprise Engineering Handbook that will provide a navigation system as well as general IA context, definitions, and references.

Each pattern will be associated with a row of the Zachman Framework. The elements of any framework "cell" (i.e., intersection of row and column) have relationships with other cells, rows, and columns. The elements of the IA view and their relationships to the rest of the Zachman Framework can be thought of as forming an "IA plane." The plane overlays the entire framework (i.e., both columns and rows) with the IA aspects of each cell, including both the model element requirements for IA and the IA services provided for model elements. The plane enables integration, because for each cell the IA concerns can be viewed alongside the non-IA concerns.

The combination of IA view and IA plane therefore allows us to achieve both separation and integration of IA. The IA view supports analyzability (to show how the system security elements will support the enterprise needs for IA) and modularized IA patterns, while the IA plane shows the integration of IA elements of the architecture with other engineering concerns. Together, the plane and the view help achieve completeness of IA concerns across the entire framework.

A more detailed perspective of how our selected IA areas relate to the Zachman views is shown in Table 1 below. This table identifies the IA elements considered at the Zachman Scope row for each IA area. Thus, there is not just one IA plane, but a separate plane for each IA area. This approach helps achieve integration of IA with engineering at a more detailed level.

Table 1: Example IA Elements at Zachman Scope Level

IA area	Elements of IA plane at Zachman scope level
Identification & Authentication (I&A)	List of functions, data, networks, and events where identification of actors is important
Access Control	List of functions, data, networks, and events important to protect from undesired exposure
Integrity	List of functions, data, and networks whose integrity (correctness) it is important to maintain
Denial of Service	List of functions, data, networks, and events important to keep available for desired use
IA Accounting	List of types of actions and events important to associate with their originating person, organization, or process

It is important to understand that the application of our results does not require the use of the Zachman set of architecture views. Other sets of views could be used, such as those in the architecture part of [RM-ODP], or those associated with the Unified Modeling Language (UML) and the Rational Unified Process [Jacobson99]. What is essential in our approach is that (1) IA be treated as a separate view, and (2) the IA view be defined and integrated across the spectrum from enterprise to implementation.

Figure 2 summarizes graphically the above discussion of the goals and approach of the research project. IA best practice is being captured and expressed in a system of patterns that forms an IA view and includes business, system, and technology level patterns. Practitioners can then apply the patterns to the engineering of specific enterprise architectures and systems.

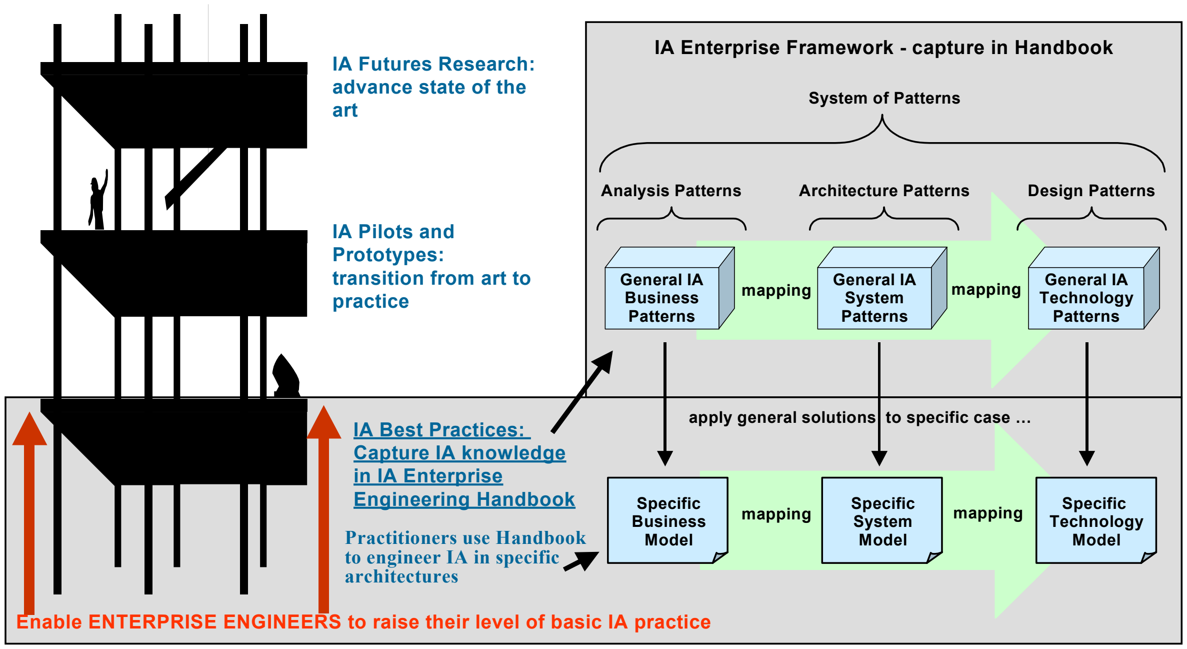
IA Futures Research: advance state of the art
IA Pilots and Prototypes: transition from art to practice
IA Best Practices: Capture IA knowledge in IA Enterprise Engineering Handbook
Practitioners use Handbook to engineer IA in specific architectures
Enable ENTERPRISE ENGINEERS to raise their level of basic IA practice
IA Enterprise Framework - capture in Handbook
System of Patterns
Analysis Patterns
Architecture Patterns
Design Patterns
General IA Business Patterns
mapping
General IA System Patterns
mapping
General IA Technology Patterns
apply general solutions to specific case ...
Specific Business Model
mapping
Specific System Model
mapping
Specific Technology Model

Figure 2. Research Project Goals and Approach

2.3 Related work

Patterns work in design and other engineering activities was identified in Section 2.2. In recent years researchers have been extending this approach to include security patterns. The Open Group is developing a system of security design patterns. A first draft is available in [OpenGroup02]. They follow the approach of the design patterns community [Gamma94]. They have defined a set of patterns organized into Entity, Structural, Interaction, Behavioral, and Available System Patterns. They have begun populating the set. In contrast to our research, their focus is on design patterns, and their classification taxonomy is less IA-specific.

A good source for information about security patterns is the web site *http://www.security-patterns.de/*. This site also has pointers to the important emerging area of security pattern languages. Examples include a language for abstract access models [Fernandez01] and a language for key management [Lehtonen01]. Schumacher and Roedig [Schumacher01a] point out that since the security field is very broad, it is important to capture relations among security patterns. Perhaps most pertinent for our research is their statement of the need for several levels of abstraction and different views, and the need to identify a suitable classification scheme. Our research directly responds to this need, not only within IA but also in the larger context of enterprise engineering.

The notion of levels of abstraction is often presented as if there were one type of abstraction. In fact, there are several types used in software systems engineering. Following [RM-ODP], we define abstraction as a simplified model with irrelevant detail suppressed. To further clarify our approach and contrast it with related work, we briefly define some abstraction types.

- Composition: A whole consists of multiple parts where each part can in turn be considered a whole consisting of multiple parts. Each whole is an abstraction of its parts.
- Generalization: A set (or type or class) may be specialized into multiple subsets (or subtypes or subclasses), each of which in turn may be further specialized. Common example: a class inheritance hierarchy. Each class is an abstraction of its subclasses.
- Categorization: A category may be instantiated, and each of the instances may be viewed as a category that can be instantiated. Common example: class vs. object. The class is an abstraction of its instantiated objects.
- Conceptualization or representation: A spectrum is defined between the end points of a problem domain (e.g., banking) and the computer. The degree or level of abstraction of a model is often defined as its conceptual distance from the computer. A common example of increasingly abstract models under this definition is: executable program (most concrete), source code, UML design, user requirements (most abstract).
- View: Any subset or concern can be defined as a view to promote separation of concerns. Examples include crosscutting concerns as in aspects, or areas of concern such as a data view, or static or dynamic view. Security/IA is generally treated as a view.

All of the above abstraction types (except Views) can be partially ordered, that is, they can be described in terms of *levels* of abstraction. These levels of abstraction are often used in conjunction with the following concepts:

- Tiers: 3-tier or N-tier architectures are usually treated as separation of concerns, but in some cases tiers are in addition regarded as separate levels or layers.
- Dependencies: A dependency chain or hierarchy may be generated from a model or component that depends on services provided by one or more other components, which in turn depend on other components. The components may be defined at different levels, typically in the context of layers.
- Layers: A layered architecture is a solution structure that typically uses a combination of the above abstraction types and related concepts. [Buschmann96] describes a "Layers" architecture pattern that depends on the structuring of component parts in a partially ordered set, grouping parts into varying levels or layers. One can use mixed-mode layering, i.e., mix different abstraction types. Fernandez and Pan [Fernandez01] use a layering approach that includes metalayer, application layer, system layer (OS/DBMS), distribution layer, and hardware configuration.

- Distribution: In a distributed system, layers can be used to support virtual peer dependencies.

The goal of achieving 'multiple levels of abstraction' is therefore somewhat ambiguous if the type of abstraction is not specified. Given the multiple abstraction types, what is important? We did not adopt the layered pattern in our approach because we believe there is too much variation and lack of clarity in its application in the community to serve as a stable context for organizing IA patterns, and because layers do not explicitly support views.

Bottom line. Our belief is that the most important elements in capturing IA knowledge are: (1) address all levels of conceptualization – not just system levels but also enterprise levels; (2) address all levels of composition from system of systems to smallest unit; and (3) capture IA patterns as a view that is applicable at each level of conceptualization and composition. The vehicle that comes closest to achieving this goal is the concept of enterprise architecture framework. Specifically, we chose the Zachman Framework for enterprise architectures, because it includes enterprise levels as well as system levels. In addition, the Zachman Framework separates views from the development or abstraction levels, and thus offers a natural basis for adding an IA view that spans those levels.

Our approach is a bit different from that of most security patterns research in two respects. One is our emphasis on the conceptual up front incorporation of IA issues at both the enterprise and system levels. At the higher levels, our IA pattern system involves decision support and choosing among alternatives. At the lower levels, we are looking to incorporate security patterns that have already been defined (see 'Related patterns in the patterns community' in our example pattern in the Appendix), rather than re-invent our own. Another difference is that most security patterns research has thus far been oriented toward capturing solutions to security problems that will be custom coded software. Our research is more oriented toward integrating modular solutions to problems into the architecture, whether the solutions are purchased as commercial off the shelf (COTS) software or hardware products, or developed locally.

Because of these differences, we are not yet certain of the extent to which existing security patterns can be plugged into our pattern system. We do, however, support the goal expressed in [Schumacher01b] of merging security patterns from the community into an integrated system of patterns. We propose the use of an enterprise architecture framework as a good context for organizing and merging patterns. We have not yet reached a conclusion as to whether different types of representation for pattern solutions are needed at different levels. For example, UML is used widely for design and for the solution part of design patterns, but it is not clear that UML is appropriate for higher level patterns, or for the decision support aspects of the patterns.

3. Research Status and Examples

Activities in our research effort thus far have included developing a taxonomy of IA areas, some of which were identified in Table 1, and using this taxonomy as a basis for identifying and organizing IA patterns. We positioned an IA view as an added column in the Zachman Framework and investigated existing patterns that relate to IA. We reviewed existing pattern templates and adapted one for our IA system of patterns. We have generated preliminary pattern trees for I&A and IA accounting, two basic taxonomy areas. We explored and captured I&A

details and incorporated them into preliminary patterns to populate the I&A pattern tree and have begun the same for IA accounting patterns.

Below we offer an example of how the area of I&A shows up in different levels of the IA view in the Framework. Figure 3 illustrates the I&A patterns that we have associated with each enterprise model (i.e., row) of the Zachman Framework. The figure also shows a preliminary pattern relationship tree. There may be multiple relationships among certain patterns, but thus far we have identified a connection between postconditions of one pattern and preconditions of another. More specifically, an arrow leading from pattern P1 to pattern P2 means that the postconditions of P1 are a subset of the preconditions of P2.

At the scope level (i.e., top row), the planning information includes enterprise level needs for identification and the types of actors in the enterprise. In the business model, the focus is on documenting the actors' geographic locations, timing schedules or cycles for the enterprise, and organizational structures that will tend to impact needs for authentication (i.e., certainty) of identities. At the system model level, designers apply I&A criteria specified at the scope and business model levels to select approaches for implementing I&A. At the technology model level the focus is on issues such as the types and numbers of authenticators and the selection or implementation of specific authentication mechanisms of appropriate types.

A preliminary draft of the I&A pattern 'Decision Tradeoffs for Automated I&A' is presented as an example (in the Appendix). This pattern is in a very simple form, and represents work in progress. For example, additional forces related to interoperability issues or to business events or business cycles need to be explored. For a complete I&A pattern set, everything will need to be defined clearly and succinctly, and related activity descriptions will need to show their interaction with I&A. After the basic patterns are defined, their relationships will have to be revisited and refined.

4. Potential benefits of our approach

Most organizations and enterprises are building architectures and using enterprise frameworks today but there is no common approach to analyzing and defining IA in these architectures and frameworks. The community does not know what to capture in enterprise frameworks and architectures for IA. If our research is successful, the potential benefits include enhancing the ability of system architects to address IA in enterprise frameworks and architectures in two ways.

First, our research results should help engineers address the need for IA early in the development process. In general, frameworks promote commonality by specifying a set of architecture products, but IA aspects have not yet been addressed. Architecture is a way to trace IA to business needs.

Second, our handbook should help to address the shortage of skilled personnel in the IA area. Today, IA staff are called upon as soon as the topic arises even though there are many basic areas that could be addressed by enterprise engineering staff. If our handbook enables system engineers to solve routine IA problems, the IA staff can be used to provide support for unique problems and to ensure all areas are addressed for completeness.

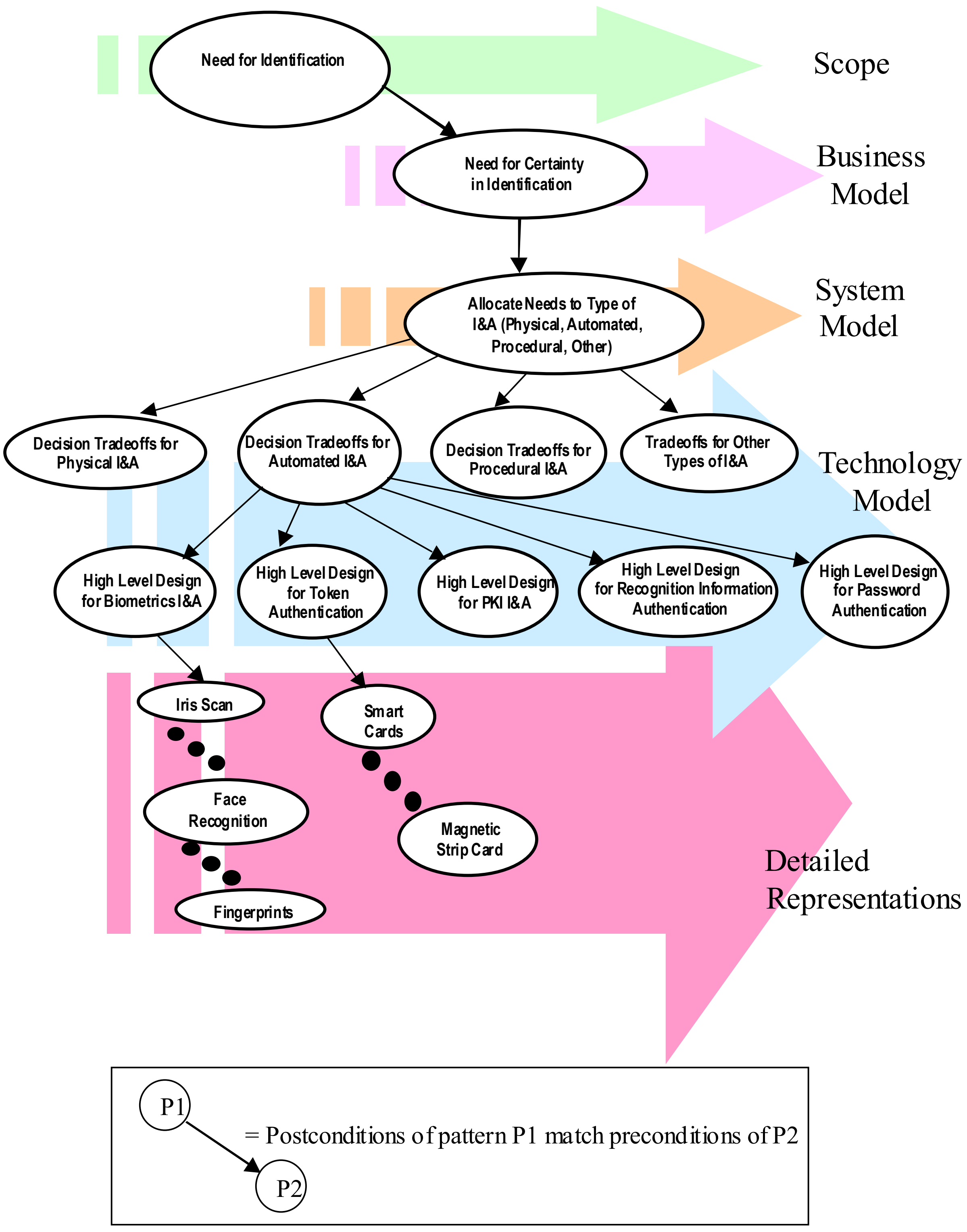
Need for Identification
Scope
Need for Certainty in Identification
Business Model
Allocate Needs to Type of I&A (Physical, Automated, Procedural, Other)
System Model
Decision Tradeoffs for Physical I&A
Decision Tradeoffs for Automated I&A
Decision Tradeoffs for Procedural I&A
Tradeoffs for Other Types of I&A
Technology Model
High Level Design for Biometrics I&A
High Level Design for Token Authentication
High Level Design for PKI I&A
High Level Design for Recognition Information Authentication
High Level Design for Password Authentication
Iris Scan
Smart Cards
Face Recognition
Magnetic Strip Card
Fingerprints
Detailed Representations
P1
P2
= Postconditions of pattern P1 match preconditions of P2

Figure 3. Preliminary I&A Pattern Tree Mapped to Framework

5. Future Plans

This research is ongoing, and our planned future work is described graphically in Figure 4.

We plan to use the pattern templates that have emerged in our work to capture the remaining patterns for I&A. Then, using what we have learned in that development process, we will complete the capture of all appropriate patterns for IA accounting and Access Control. Throughout that development process, we will be mapping back to the Zachman Framework to ensure that our coverage achieves the completeness that will be needed. As we proceed, we will also be capturing the patterns, their interrelations, and guidance on their use, in the IA Enterprise Engineering Handbook. While this first pass at a "living" handbook will not address all the areas of the IA taxonomy, we believe it will be a sufficient body of work that will encourage its continued evolution.

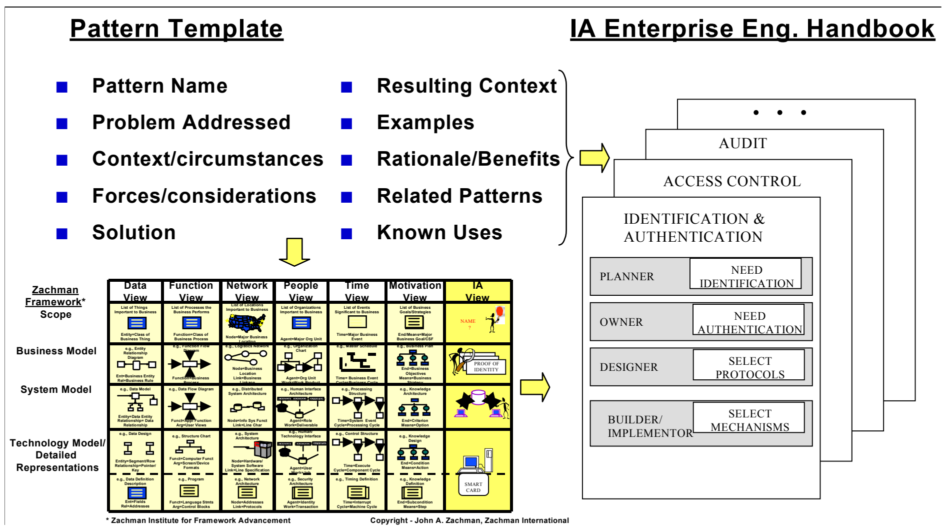

Figure 4. Future Plans: A Framework-Based Handbook of Patterns

A larger case study will be undertaken using the integrated IA handbook on an existing MITRE effort allowing us to determine the usability of the IA system of patterns created. It will also provide an opportunity to identify any shortfalls in the IA view or handbook that can be adjusted.

Appendix A. Draft I&A Pattern

Pattern: Decision Tradeoffs for Automated I&A

Problem Addressed

Once the decision is made to allocate specific I&A requirements to automation (i.e., to enterprise information systems), the builder must decide among a number of alternatives for implementing those needs as appropriate across the enterprise's systems using one or more types of automated I&A approaches. The strategies addressed in this pattern cover both identification and authentication. Therefore, the term "I&A" is used.

Context (Pre-condition)

This pattern addresses those enterprises where there is a need for identification and authentication via an automated solution. This pattern shows the builder how to select an appropriate automated I&A solution based on the need for reliability of identification and other forces, derived from the business and system models, that affect the selection.

This pattern applies when the following preconditions have been met:

- The need for identification has been established for actors interacting with specific enterprise business processes. (Pattern: Need for Identification)
- The need for reliability in this identification has been established (Pattern: Need for Certainty in Identification), expressed as the cost (monetary, goodwill, or other) to the enterprise if :
    - A business process accepts a false identity; that is, one that does not match the actual actor.
    - A business process refuses to accept a legitimate identity; that is, one that matches the actual actor.

  This cost is usually referred to as the cost of compromise with respect to I&A.
- Some (perhaps all) of the identified I&A requirements have been allocated to automated I&A processes associated with specific information systems that support specific business processes. (Pattern: Allocate Needs to Type of I&A).
- Geographic distribution of the information system access points and the information system I&A needs at the various locations have been determined. (Pattern: Allocate Needs to Type of I&A)

Forces

To select among individual and combinations of automated I&A types, the builder will need to reference and balance the following forces at a minimum. Other forces unique to the enterprise may also need to be considered.

Force	Components of Force	Examples
Per-user overhead	Per user cost to the enterprise	Cost of SecureID token
	Initiation cost to the user	Users' ability to remember additional password(s)
Per-use cost	Physical risk, discomfort, or effort by user	Retinal scan damage
	Time delay to perform	Slow confirmation of certificate revocation status
Per entry point cost	Special hardware devices	Cost of smart card reader
	Special software	Cost of applications enhanced to deal with PKI
	Physical or procedural controls at entry point	Guard at front desk to office
General overhead and management costs	Security administration	Service to initialize passwords
	Registration	Service to establish user identity
	Operations and Maintenance costs for special hardware or software	Maintenance for iris scanners at all system entry points
Reliability	Percentage of false positives	Simple passwords are easily stolen
	Percentage of false negatives	Complex passwords are easily mis-typed; can combine with three-strikes-you're-out to lock users out
Cost to the enterprise of authenticator compromise	Cost of replacing an authenticator if it is lost or stolen Explanation: An untrusted or unauthorized actor obtains the identification and authenticator of a user.	The cost of establishing a new password is small. However, the cost of dealing with a compromised biometric authenticator, such as a stolen retinal image together with a technique for successfully using it to get past an entry point, can be quite high. A valid user may be permanently barred from using the system or the authenticator approach may need to be replaced.
User base constraints	Sophistication	Unsophisticated users may find PKI difficult to use
	Scale	Large user base interacts with per-user cost to the company
	Trustworthiness	Likelihood users will report lost tokens

Force	Components of Force	Examples
	Handicaps	Quadriplegics may be unable to type passwords
	Includes humans, software programs, or both	Software programs may act as agents for users or need access to enterprise resources for routine system housekeeping functions (e.g., e-mail delivery)
	User mobility Explanation: Users do not remain in a single location or always use the same equipment or entry point	Employees need to be able to access the corporate intranet or administrative systems from any workstation at any enterprise site
User equipment constraints	Controlled by user, enterprise, or third party	User may need to authenticate to an enterprise system from a shared computer at trade show
	System services (i.e., system platform) variability	Operating system may be Mac, UNIX, Windows
Enterprise geographic constraints	Self-contained computing Explanation: All users wishing to access system resources must be physically present at the system location	To gain access to the system, all users must first gain access to a single physical location that is protected by physical I&A mechanisms
	Distributed computing Explanation: Systems share resources across multiple enterprise sites or across distinct locations within a site; includes distributed systems	Employees can gain access to the corporate Intranet through workstations at any corporate site
	Mobile computing (includes laptops and wireless devices)	Employees need to access enterprise resources from cell phones
	Telecommuting Explanation: Employees need to access enterprise resources from home systems or other, non-company controlled systems	Employees need to access corporate resources from workstations at shared, third party maintained telecommuting hub offices
	World wide computing Explanation: Enterprise has overseas offices that share computing resources	Users need access corporate resources from locations with unreliable or poor quality communications services
Infrastructure constraints	Middleware/system services	I&A process adds requirements to the functionality of a DBMS that will be used to store identification or authentication information

Force	Components of Force	Examples
	Processing power	PKI encryption requirements add substantially to the processor load
	Data storage	I&A process adds substantially to enterprise automated storage requirements
	Hardware configuration	Token reader requires extra or unusual hardware port on workstations/system consoles
	Network reliability and availability	Authentication process requires access to remote or third party resources (e.g., database)
	Network bandwidth	Authentication process adds substantially to network bandwidth requirements
	Software development	PKI requires specific functionality to be included in all applications

Solution

To determine the I&A type(s) that will meet the enterprise's needs, these steps are followed:

- From the Forces table, determine what forces apply to the enterprise
- Refer to the table of I&A type characteristics below to find a solution that most closely fits the enterprise combination of selection criteria tradeoffs and forces.[1]

Solution option	Criteria indicating selection	Characteristics
UserID/ password	Cost of compromise is low Number of users is large Number of different passwords per user is low	Low per-user cost Low management cost Moderate reliability Impact on infrastructure is minimal
Hardware token	Cost of compromise is moderate to high Adequate password option requires long, difficult passwords	Moderate per-user cost Moderate management cost Moderate to high reliability, especially if

[1] Note: we do not associate the types of I&A with any form of security/assurance levels. The actual use of the I&A type in a given system will vary widely and impact any such "levels." That is, a strong I&A type applied in a weak manner will not offer any increased security or assurance.

Solution option	Criteria indicating selection	Characteristics
	or multiple passwords per user Entry point is less secure No software actors User base is mobile Not suitable for some types of mobile computing (e.g., cell phones)	combined with password for use of token Moderate to high costs per entry point
Biometrics	Cost of compromise is high to enterprise and user Entry point is insecure (physically) Tokens or passwords are not acceptable No software actors Not suitable for some types of mobile computing (e.g., cell phones)	Moderate to high per user cost Variable management cost (depending on the type of biometric selected) Potential for high reliability (depending on the type of biometric selected) Moderate to high infrastructure impacts (cost of additional processor, storage, and network loads) High cost per entry point Successful identity theft has the potential for severe problems for a user
Additional User-Recognition information	Cost of compromise is low Very large user base (unwieldy for normal registration processes) No software actors No requirements on entry points or user equipment	Low per user cost Low management cost Moderate to low reliability Minimal infrastructure impacts
PKI	Cost of compromise is high Software actors must be supported	Moderate per user cost Moderate to high management costs (a trusted third party is usually involved) Cost per entry point is low Infrastructure impact is very high (software development practices are impacted) High reliability Suitable for mobile computing

Resulting Context (Post-conditions):

If the pre-conditions are met, then use of this pattern allows traceability from business needs to the type of automated I&A approaches selected.

The pattern results in the identification of the type(s) of automated I&A to be employed for enterprise information systems. The pattern result identifies where in the enterprise systems the I&A type(s) appear. Considerations for their use characteristics (e.g., token PIN length, false positives and negatives acceptable scope for biometrics) are identified for the further requirements of the system components.

It is possible and even likely that multiple types of automated I&A result from applying this pattern. In those cases, the selected types have patterns that can be applied, and all the selected types must be integrated as part of the engineering process.

Example

This is an example for where "Additional User-Recognition Information" makes sense as the selected I&A type. The user base for the publicly accessible taxpayer information system at the IRS (i.e., the system that supports taxpayer queries over the web) is extremely large and unwieldy. The amount of confidential taxpayer information available through this system is limited. Therefore, standard login identification will be used with additional user-recognition information as an increased authentication mechanism. In this case, the recognition information to be used is adjusted gross income. Using this type of I&A minimizes the requirements for infrastructure support, since this recognition information is already stored in IRS systems. As a general rule, the I&A information is not revoked. Specifically, the selected recognition information is not revocable. Malicious users are denied access by removal of their identification, with maintenance of that information to prevent re-introduction of those users.

Related Patterns

Related Patterns in our system:

- Registration pattern for registration of identities is a normal precursor to I&A
- Revocation pattern is used to remove an identity that has been established by I&A
- Other I&A patterns
    - Identify Need for Identification
    - Identify Need for Certainty in Identification
    - Allocate Needs to Type of I&A (Physical, Automated, Procedural, Other)
    - The following can be applied depending on the automated type(s) selected in this pattern:
        - High Level Design for Passwords
        - High Level Design for Tokens
        - High Level Design for Biometrics
        - High Level Design for Recognition Information
        - High Level Design for PKI

Related Patterns in the patterns community:

- Authenticator pattern [BrownF99] is used to implement authentication for remote objects

- The Open Group has defined an authenticator API [OpenGroup01] that provides a high-level generic model to use for human authentication, with particular emphasis on biometric technology.
- Key management language [Lehtonen01] has patterns useful for PKI
- Password patterns [Riehle02] consist of a series of heuristics on selecting and using passwords.

References

[BrownF99] F. Lee Brown, Jr., James DiVietri, Graziella Diaz de Villegas, and Eduardo B. Fernandez, "The Authenticator Pattern," in PLoP Proceedings 1999.

[BrownW99] William J. Brown, Hays W. "Skip" McCormick III, and Scott W. Thomas, *Anti-Patterns and Patterns in Software Configuration Management*, John Wiley, 1999.

[Buschmann96] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal. *Pattern-Oriented Software Architecture - A System of Patterns*. Wiley, 1996.

[CC99] Common Criteria Implementation Board, *Common Criteria for Information Technology Security Evaluation,* Version 2.1, 1999 (also ISO IS 15408). Available at http://csrc.nist.gov/cc/ccv20/ccv2list.htm

[Cheswick94] Cheswick, W. and S. Bellovin, *Firewalls and Internet Security*, 1994, Addison-Wesley Publishing Company, Reading, MA.

[Dikel00] David M. Dikel, David Kane, and James R. Wilson, *Software Architecture: Organizational Principles and Patterns*, Prentice Hall, 2000.

[DoD85] DoD, *Department of Defense Trusted Computer System Evaluation Criteria*. DoD5200.28-STD. Washington, DC, DoD, 1985.

[Fernandez01] Eduardo B. Fernandez and Rouyi Pan, "A Pattern Language for Security Models," in PLoP Proceedings 2001.

[Fowler96] Martin Fowler. *Analysis Patterns: Reusable Object Models*. Addison-Wesley, 1996.

[Gamma94] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley, 1994.

[Jacobson99] Ivar Jacobson, Grady Booch, and james Rumbaugh. *The Unified Software Development Process*. Addison-Wesley, 1999.

[Lehtonen01] Sami Lehtonen and Juha Pärssinen, "A Pattern Language for Key Management," in PLoP Proceedings 2001.

[NCSC85] National Computer Security Center. *Computer Security Requirements -- Guidance for Applying the Department of Defense Trusted Computer System Evaluation Criteria in Specific Environments*. CSC-STD-003-85, Fort George G. Meade, National Computer Security Center, 1985.

[NIST01] National Institute of Standards and Technology, "Engineering Principles for Information Technology Security (A Baseline for Achieving Security) ," NIST Special Publication 800-27, June 2001.

[NIST94] National Institute of Standards and Technology, "Security in Open Systems," NIST Special Publication 800-7, July 1994.

[NIST95] National Institute of Standards and Technology, "An Introduction to Computer Security: The NIST Handbook," NIST Special Publication 800-12, October 1995.

[NIST96] National Institute of Standards and Technology, "Generally Accepted Principles and Practices for Securing Information Technology Systems," NIST Special Publication 800-14, September 1996.

[NIST-NSA92] National Institute of Standards and Technology and National Security Agency, *Federal Criteria for Information Technology Security*. Vol. 1 Protection Profile Development, Vol. 22 Registry of Protection Profiles, Version 1.0, Gaithersburg, MD, National Institute of Standards and Technology/Computer Systems Laboratory, 1992.

[NRC99] Fred B. Schneider, ed., *Trust in CyberSpace*, The National Research Council, Washington, 1999.

[NSTISSI99] *National Security Telecommunications and Information Systems Security Instruction* (NSTISSI) No. 4009, January 1999

[OpenGroup01] The Open Group, CDSA/CSSM Authentication: Human Recognition Service (HRS) API, Version 2, *Open Group Technical Standard*, June 2001.

[OpenGroup02] The Open Group, *Guide to Security Patterns.* Draft 1, 5 April 2002.

[Pipkin97] Pipkin, D. L., *Halting the Hacker*, 1997, Prentice Hall PTR, Upper Saddle River, NJ.

[PL74] Public Law 93-579. *Privacy Act of 1974*. Title 5, US Code, Section 552a, 1974. Available at http://www.fs.fed.us/im/foia/patxt.htm.

[Riehle02] Riehle, Dirk. Password Patterns. At http://www.riehle.org/technical-forum/wiki.cgi?PasswordPatterns.

[RM-ODP] International Organization for Standardization (1995) *Basic Reference Model of Open Distributed Processing*. ITU-T Recommendation X.902 | ISO/IEC 10746-2: *Foundations* and ITU-T Recommendation X.903 | ISO/IEC 10746-3: *Architecture*.

[Schumacher01a] Markus Schumacher and Utz Roedig, "Security Engineering with Patterns," in PLoP Proceedings 2001.

[Schumacher01b] Markus Schumacher, "Merging Security Patterns," in PLoP Proceedings 2001.

[Sowa92] J.F. Sowa and J. A. Zachman. "Extending and Formalizing the Framework for Information Systems Architecture." *IBM Systems Journal*, vol. 31, no. 3, 1992. IBM Publication G321-5488.

[Zachman87] John A. Zachman. "A Framework for Information Systems Architecture." *IBM Systems Journal*, vol. 26, no. 3, 1987. IBM Publication G321-5298.

[ZIFA02] Zachman Institute for Framework Advancement (Web site). http://www.zifa.com/