2014 47th Hawaii International Conference on System Science

Towards a Design Theory for Trustworthy Information Systems

Leslie J. Waguespack, David J. Yates & William T. Schiano
Bentley University
Waltham, Massachusetts, USA
lwaguespack@bentley.edu *dyates@bentley.edu* *wschiano@bentley.edu*

Abstract

The lack of a competent design theory to shape information system security policy and implementation has exacerbated an already troubling lack of security. Information systems remain insecure and therefore untrustworthy even after more than half a century of technological evolution. The issues grow ever more severe as the volume of data grows exponentially and the cloud emerges as a preferred repository.

We aspire to advance security design by expanding the mindsets of stakeholder and designer to include a more complete portfolio of factors. The goal of security design is to craft choices that resonate with stakeholders' sense of a trustworthy system. To engender trust, security must be intrinsic to any definition of IS design quality. Thriving Systems Theory (TST) is an information systems design theory focused on reconciling and harmonizing stakeholder intentions. Formulating security design through TST is a starting point for a quality-based security design theory for trustworthy information systems.

1. Introduction

In 2002 the OECD reiterated its call for awareness and action with the "growing number and wider variety of threats and vulnerabilities" that mark the Internet age of computing [36]. In those guidelines OECD declared:

Only an approach that takes due account of the interests of all participants, and the nature of the systems, networks and related services, can provide effective security. [36]

The first steps towards a design theory for trustworthy information systems reported herein respond to that OECD call by framing an information system security design through Thriving Systems Theory [50, 51] that maps TST's design choices onto information system policies, mechanisms, resources, users, roles, and assurance. No aspect of information systems design is impacted more by the evolving environment of hardware and software technology, connectivity, distributed storage and computation than is security, further confounded by societal, economic, political, and national security pressures [4]. Even after more than half a century, information systems remain insecure and therefore untrustworthy. In many ways, gaps in security are more severe than ever as more data and systems are moved into the cloud [25: p. 14]. Security concerns continue to increase with the number and variety of systems that connect to the Internet. Data and software services are moving online as part of the move toward cloud computing [15]. Industrial, medical, and scientific devices are also migrating their monitoring, control, and data management functions to the Internet. At the same time, the number of people online (including careless or malicious users) and perverse economic incentives that encourage attacks on information systems are both increasing. These and other trends are combining to make trustworthy computing more important than ever.

Lack of design thinking about security has led to many gaps in information systems security. This paper is a first step in the development of a design theory for trustworthy information systems. We explore how we can increase our understanding of system security through design thinking based on a design theory. We need principles of form and function in a theory to make it easier for stakeholders to consider security as an integral design dimension of their information systems and business models [23, 24, 52]. The design thinking that follows provides a way to understand a massively complex problem space and consider design choices to improve the fundamental trustworthiness of information systems.

Stakeholders include owners, managers, designers, clients, and users – the entire community involved in service delivery. Each stakeholder presents a distinct aspect of reliance on the system's behavior [11]. Stakeholders' trust in information systems is driven by a combination of 1) responding to the stakeholders' tacit expectations and 2) shaping those expectations by crafting a security model that defines trustworthy system behavior and outcomes. The experience of trust in an information system results from the policies and mechanisms that are intended to fulfill the security requirements of the system and the stakeholders' facility with them [14].

Security policies and mechanisms refer to users, roles, and resources in the context of an information

978-1-4799-2504-9/14 $31.00 © 2014 IEEE
DOI 10.1109/HICSS.2014.461

system. In the design theory language of Gregor and Jones [23], threats, policies, mechanisms, resources, users, roles, and assurance are the core constructs a design theory for trustworthy information systems must address.

Security assurance, or simply assurance, is confidence that an entity meets its security requirements, based on specific evidence provided by the application of assurance techniques. [10: p.481]

That *confidence* must extend beyond the security mechanisms implemented in software and hardware to include policies assuring that they resonate with stakeholder intentions. We propose a theory of design that shapes both the security requirements and the implementation to achieve the stakeholders' trust.

The paper begins by arguing that security needs a design theory and describing the benefits this would yield. There follows a brief description of Thriving Systems Theory and its fifteen choice properties that mediate stakeholder experience and design quality. We describe the trustworthiness of a system as the whole of the security choices evidenced by the choice properties that resonate with stakeholders. Policies and mechanisms from the security literature form a taxonomy grouped by choice property. They illustrate how security design choices shape the stakeholder(s)' experience of *trustworthiness* mediated through the lens of the choice properties. The paper concludes with a summary of findings and also discussion of limitations and future work.

2. Information system security needs a design theory

Designing security for information systems has been particularly challenging since the technologies that make up these systems, e.g., operating systems [10], databases [6], networks [47], and the world-wide web [20], have traditionally used different models for security. Furthermore, the communities of research and practice that develop these technologies do a poor job of learning from each other's best practices in designing for security.

It is well known in the operating system community that is it necessary to distinguish the level of privilege required to run code from that required to read a file (or access other resources). Furthermore, before code is run on a system, the originator of the code should be authenticated and authorized or acknowledged as trusted by the user. The design principles behind such fundamental security mechanisms have been articulated for more than three decades [6, 16, 41]. Yet the database development and operations communities have allowed SQL injection attacks to cause significant damage to businesses and consumers (e.g., reputation damage, financial loss, identity theft, etc.) [7] by not properly observing these design principles. The development, networking, and operations communities have made similar mistakes, which tend to garner less publicity. By not thinking of routing advertisements as parameters to code that modify critical shared resources – routing tables on the Internet – service providers have frequently allowed information that is transported over the Internet to be misrouted and delivered to a hacker's network or machine [47].

All informed security communities, however, understand that security benefits from following an appropriate design process in the context of a system lifecycle [12, 18]. One goal of our research in this area is to develop a design theory for information systems security so that communities developing and operating different information technologies can share knowledge and best practices using a common frame of reference. By informing each other's design practices with a sound design theory, the outputs of these communities – products and services that make up information systems – will combine to create more trustworthy systems, with fewer vulnerabilities and an improved stakeholder sense of security and welfare.

3. Thriving Systems Theory

We choose Thriving Systems Theory (TST) as a framework to pursue a design theory of trustworthy information systems. TST rests on three pillars of theory: Christopher Alexander's *living structure* in *The Nature of Order* [2], Lakoff and Johnson's cognitive-linguistics and conceptual metaphor that explain human understanding and perception [28], and Fred Brooks' *essence* and *accidents* in systems development [13]. As security design cannot be separated from information system design we shall focus on system security aspects guided by TST's theoretical foundations. A full exposition of TST is found in [50].

3.1. Christopher Alexander's living structures

The heart of Thriving Systems Theory is a set of properties that explain the sense of resonance that someone experiences when encountering an artifact. The properties stem from Christopher Alexander's theory of *living structures* based upon the human perception of *order* inherited through biological and sociological evolution. Alexander identifies fifteen properties that evince a sense of order in the observation of art or architecture experienced as a confluence of property affects. Every design choice exhibits the fifteen properties in varying intensity. A choice may exhibit some properties intensely while other properties are all but

imperceptible. In his theory, the ultimate confluence presents as a degree of *wholeness* that a stakeholder perceives in the design – a level of satisfaction, design quality.

3.2. Physical structures to information systems

TST translates Alexander's properties (a vocabulary grounded in physical space, geometry, color, and texture) as *choice properties* in an abstract realm of systems and models of systems [50, 51]. (These choice properties appear in Table 1, side-by-side with their associated design action and definition.) The fifteen choice properties characterize the perceived organization in systems that in confluence resonate with our human conception of order. Each choice property explains a distinct cognitive lens through which humans recognize ordered-ness that conforms to an innate conception of *living structure* as Alexander defines it. Constructs, images, and concepts that reflect this form resonate with stakeholders. They facilitate recognition and understanding that engender satisfaction as the experience conforms to some extent with stakeholders' expectations. Design quality increases as design actions shape (and reshape) choices that better resonate with stakeholder expectations [50].

3.3. Cognitive linguistics and metaphor

TST's treatment of qualitative stakeholder experience proceeds from Lakoff and Johnson [28]: 1) human perception is mediated by innate conceptual metaphors through which we recognize ordered-ness, 2) the transmission of ideas through any form of human communication is imperfect and therefore all communication is metaphorical, and 3) any conception of reality is incomplete therefore satisfactory communication relies on conscious and careful abstraction [50].

"Everything that can be counted does not necessarily count; everything that counts cannot necessarily be counted." – Albert Einstein

TST is vested in quantifiable and qualitative stakeholder experience. Aspects of design appear simpler when reduced to numbers. But human satisfaction relies on individual perception that encompasses the aesthetic – tapping into the stakeholder's sense of value in the artifact. TST focuses on design elements that convey ordered-ness as a fundamental stimulus of stakeholder satisfaction, design quality.

3.4. Essential and accidental design choices

An artifact (e.g., an information system) is the expression of a concept; as such it is a metaphor. Brooks' *essence and accidents* provide a means to distinguish the metaphor's design elements that are intrinsic to it as concept from those that are accidental in its representation (i.e. implementation) [13]. Any artifact of design is the rendering of both the elements intended to be intrinsic and those that are accidental.

Table 1. TST Choice Properties

Choice Property	Design Action	Action Definition
Modularization	modularize	employing or involving a module or modules as the basis of design or construction
Cohesion	factor	express as a product of factors
Encapsulation	encapsulate	enclose the essential features of something succinctly by a protective coating or membrane
Composition of Function	assemble	fit together the separate component parts of (a machine or other object)
Stepwise Refinement	elaborate	develop or present (a theory, policy, or system) in detail
Scale	focus	(of a person or their eyes) adapt to the prevailing level of light [abstraction] and become able to see clearly
Identity	identify	establish or indicate who or what (someone or something) is
Patterns	pattern	give a regular or intelligible form to
Programmability	generalize	make or become more widely or generally applicable
User Friendliness	accommodate	fit in with the wishes or needs of
Reliability	normalize	make something more normal, which typically means conforming to some regularity or rule
Correctness	align	put (things) into correct or appropriate relative positions
Transparency	expose	reveal the presence of (a quality or feeling)
Extensibility	extend	render something capable of expansion in scope, effect, or meaning
Elegance	coordinate	bring the different elements of (a complex activity or organization) into a relationship that is efficient or harmonious

In TST an artifact that thrives is marked by the stakeholders' comprehensive understanding of the artifact's environment, its ecology, and the prospects for the evolution of both [50]. The distinction between essence and accident guides the designer's decisions to

reshape accidental design choices to improve stakeholder facility while preserving essential choices that define the artifact's purpose as the stakeholders understand it.

3.5. Thriving Systems Theory as an ISDT

Aligned with Gregor and Jones' anatomy of a design theory [23], TST defines design quality as the resonance between stakeholder(s) and artifact. Gregor and Jones define eight components of design theory [23]. Using their framework, Thriving Systems Theory is a design theory comprising: a) the conceptualization of an artifact as a system of choices, b) design quality characterized as satisfaction, c) satisfaction experienced as the alignment and resonance between stakeholder expectation(s) and the artifact, d) choice properties as the taxonomy of assessable referents of stakeholder experience, and e) corresponding design actions strengthening choice property intensity [51]. The choice properties in Table 1 embody the components of artifact mutability and the principles of form and function identified in Gregor and Jones [23].

As a design theory, TST equips designers to purposefully craft choices to manipulate the relative strengths of choice properties; targeting a resultant resonance, the sense of satisfaction that stakeholders will experience with the artifact.

4. Framing trustworthiness in Thriving Systems Theory

As an information systems design theory, TST designates *design choice* as the focal construct [51]. In order to apply TST, design choices must be grounded in the context of a specific artifact so each choice property assumes a range of design alternatives and interrelationships. Table 2 illustrates aspects of security design as constructs of trust.

Our focus on security in information system design grounds choices in a context of threats, policies, mechanisms, resources, users, roles, and assurance that form a *security model* expressing the stakeholder(s) conception of a trustworthy information system. Table 2 lists fifteen aspects of security design choice grouped by the pronounced choice property they entail. These aspects are drawn from the security literature. Although we cannot claim these aspects are comprehensive, they are a representative compendium of information system security principles and protocols in academic and industrial practice. We use them here to represent the available design decisions that would fully populate a mature security design theory – the *principles of implementation* in [23].

The goal of security design is to craft design choices that resonate with the stakeholders' sense of a system that fulfills their expectations for trustworthy behavior – satisfying their expectations for functionality but also sustaining a sense of freedom from undesirable system behavior or misuse.

Table 2. Security Design Aspects

Choice Property	Security Design Aspect
Modularization	**Defining Domains**: a topological definition of protection by requirement where constituent elements are subject to consistent policy and protection mechanisms
Cohesion	**Simple Trusted Components**: a preference for atomic protection mechanisms and system elements
Encapsulation	**Separation**: segregating protection domains and mediating their exchange of information, control and authority
Composition of Function	**Linking Roles & Domains**: cascading authentication and separation of domains to attenuate privileges
Stepwise Refinement	**Defense in Depth**: graduated protections in layers spanning application, platform and communication architecture
Scale	**Least Privilege**: preferring that domain access spans the minimum range feasible to support required functionality
Identity	**Identity Management**: comprehensive and definitive naming of system elements to allow application and assurance of security mechanisms
Patterns	**Few Trusted Components**: minimal and symmetric formulation of criteria, privilege and protection across domains
Programmability	**Authorizing Operations**: the ability to adjust the scope and depth of protection to meet stakeholder security concerns
User Friendliness	**Manageable Access**: coherent and user-accessible policy and protection mechanisms to manage and monitor domains
Reliability	**Complete Mediation**: assured system-wide application and enforcement of protection mechanisms
Correctness	**Assurance**: evidence based monitoring of policy and protection mechanisms across domains
Transparency	**Auditing**: facility for threat identification and classification supporting forensics and ongoing policy review and evolution
Extensibility	**Risk Management**: dynamic policy and protection specification supporting timely response to the changing threat landscape and evolving stakeholder intentions
Elegance	**Elegance**: protection mechanisms effectively, efficiently, and simply organized, realizing a security policy resonating with the stakeholder community's conception of security and welfare

4.1. Aspects of design entailing security

Quality in TST is expressed through the confluence of choice properties. Stakeholders experience the af-

fects of some or all of the properties in every design choice. The following aspect elaborations (4.1.1 – 4.1.15) illustrate security design choices that entail a particular property in that confluence. Each security aspect elaboration is intended to characterize rather than enumerate design choices [43]. Each elaboration is prefaced with the design actions that shape a choice to intensify that TST choice property [50]. (The first reference of each preface is to the design actions as defined in TST and the second is to the term's definition in the computing literature.) Each aspect characterizes choices that may shape both security model requirements as well as implementation protocols. The choice properties thus constitute a framework informing the design of security policy and the design of system features to realize that policy.

4.1.1. Defining Domains entailing *modularization*.

*… **modularization** is to partition and associate system knowledge that facilitates "divide and conquer" problem solving or the segmented exposure of system features aligned to the stakeholders' intention. The model designer's tasks are to compartmentalize, to aggregate and to express the system as wholes and parts. And in so doing the whole is revealed through "bite-sized" pieces that promote comprehension and facilitate management.* [50: p. 30, 5]

Security domains organize the whole as a collection of parts while decomposing the problem space and separating concerns. They identify similar concerns while enumerating and designating consistent treatments. Domains group users, roles, and resources needing the same security. Domains may reflect distinctions imposed by stakeholders as well as innate characteristics of constituent elements. Domains reflect a divide-and-conquer design strategy. [41: p. 1300, 30: p. 8]

4.1.2. Simple Trusted Components entailing *cohesion*.

*… **cohesion** results from factorization that reveals/defines the fundamental system features. The model designer's tasks are to identify the fundamental, to distill the idea, to isolate the concept, to separate and distinguish the part, to name the primitive, to minimize coupling and to define the elemental component. And in so doing the model designer renders the choice individually and distinguishably complete in its own purpose.* [50: p. 29, 55]

Atomic security mechanisms and system components that minimize complexity and reduce testing effort in assurance are easier to document, explain, understand, catalog, and reuse because their interactions are more predictable [29]. Trusted component assurance is costly and a disincentive to thorough praxis. Simplifying trusted component design and paying careful attention to defining domains can control cost and improve quality. [30: p. 7]

4.1.3. Separation entailing *encapsulation*.

*… **encapsulation** results from identifying and insulating essential features while controlling access through disciplined, contractual interfaces. The model designer's tasks are to protect, to compartmentalize, to relegate, to steward, to cast, to recast, to virtualize, to package, to mask, to portray, to "component-ize," to characterize, to abstract and to hold inviolate. And in so doing not only are the internals "protected," but the choice's clients are freed from any obligatory knowledge of the details of the choice's internals.* [50: p. 30, 46]

Confidentiality and integrity rely on permitting interaction only with appropriate authorization. This requires the definition and mechanisms for domain separation (resources as members of a domain and/or individually). Interaction is only secure if separation assures unauthorized interaction is interdicted [40]. Effective separation usually rests in part on host level security mechanisms (e.g. memory and process management). Separation reifies domains by materializing the boundaries that they define. [41: pp. 1282-1286, 10: pp. 444-448]

4.1.4. Linking Roles & Domains entailing *composition of function*.

*… **composition of function** exploits the constituent potential of model choices combining them as individual contributors to a new and distinctive configuration. The designer's task is to build, to compose, to manufacture, to piece together, to assemble, to construct, to combine, to package, to fabricate, to erect, to connect or join. And in so doing the designer exercises the constituent parts' potential deposited through extensibility and/or programmability by producing a whole that is "greater than the sum of its parts."* [50: p. 32, 32]

The essence of computer security is the authorized association of an authenticated user (or a user's agent) with a protected resource – sometimes information and sometimes function. Domains that define users, resources and privileges are intrinsic to this apparatus. Roles can organize the authority for domain access distinct from user identity and enable authorization-by-responsibility [42]. They compose collections of authority and offer the flexibility to manage intersections among them. [10: pp. 43, 353, 30: p. 8, 41: p. 1295, 31]

4.1.5. Defense in Depth entailing *stepwise refinement*.

*… **stepwise refinement** is the exposure of system features in digestible increments in a spiral of incremental explanation as in a pedagogy. The model designer's task is to deliver a succession of reinforcing representations that explain the parts within an outline of the whole, an elaboration of system elements that shapes the observers' understanding consistent with the system's stakeholder intentions and an exposition of structure as parts assembled to form the whole. In a sense strengthening this property is analogous to revealing the observable refinements evidenced in evolution – a succession of steps that has led it now to what it is.* [50: p. 29, 9]

A security policy may be interpreted or implemented differently in different contexts and at distinct levels of architectural granularity. Nested security domains make it possible to match graduated degrees of risk with commensurate levels of protection effort. The strategy may include varying security protocols at domain boundaries to increase threat deterrence or require more authentication effort, thus minimizing the potential of "penetrating" nested security domains. [35, 41: p. 1282, 30: p. 7]

4.1.6. Least Privilege entailing *scale*.

*… **scale** results from the imposition of a telescoping sense of focus that may be directed to an observer's purpose and renders in clarity the system features relevant to that purpose. The model designer's tasks are to direct attention, to highlight, to draw attention to, to lend perspective, to acquaint, to draw parallels with, to contextualize, to put into perspective, to lead an observer through the unfolding, to familiarize, to introduce, to bring into focus, to zero in, to target and to "point to." And in so doing the model designer provides a telescoping granularity of comprehensibility to suit the requirements of a variety of observers.* [50: p. 33]

Access and privilege should be responsibility-based and assigned to be minimally sufficient to achieve a responsibility. Protection and privilege should expand and contract with responsibility. There should be a synergy between this design characteristic and roles since access and privilege should result from agents' roles in the mission of the system rather than from their identities. [41: p. 1281, 30: p. 7, 10: p. 125]

4.1.7. Identity Management entailing *identity*.

*… **identity** results when a modeling element is named and its existence is recognized. The naming of model elements constitutes the vocabulary describing and explaining the whole. To "name" something is to "know" it, to distinguish it among the rest, to justify its individual existence, to recognize its distinctiveness, to carve out a subset of the universe and label it, to collect its attributes and package them as a definite concept. To establish element identity is at the core of language; that a "name" can take the place of all that is known about an element and carry that knowledge through an explanation or analysis. A "name" may be completely distinguishing or categorical respectively expressing either individuality or a shared context. In any case a "name" is a handle with which to grasp and carry a concept within a conversation be it noun (subject), verb (predicate) or adjective (modifier).* [50: p. 32, 27]

Authorization depends on identification. Unless identity can be trusted, the association of an agent with a resource cannot be trustworthy [53]. Even when authority is delegated through roles to achieve association with resources, authentication is still needed (i.e. in assuring *confidentiality*). Resources and other elements also require an unambiguous, discrete identity to assure that their assignment to a domain or access by an agent is not confused by subterfuge or accident (i.e. in assuring *integrity*). [41: p. 1285, 30: p. 6]

4.1.8. Few Trusted Components entailing *patterns*.

*… **patterns** results from discovery and/or designation of explicit similarity and difference. The model designer's tasks are to characterize similarity, to expose repetition, to map consistency, to reuse the familiar, to prescribe the evolution, to weave, to interlock, to establish a rhythm, to facilitate a path of lesser resistance, to foreshadow, to anticipate, to lead, to invite, to predict, to train and to condition. And in so doing the pattern strength of the choice both invites and conditions the observers to consider reuse.* [50: p. 34, 19]

A synergy between risk assessment and protection design enables effective, cost/benefit analysis. Standardizing security criteria and protocols around comparable risk allows centralization of security apparatus. Pursuing standards encourages consistent risk awareness among stakeholders. Vulnerability can be compartmentalized behind standardized security protocols consistent with resource criticality. Compartmentalization facilitates a systematized strategy for consolidating "trust mechanisms" and minimizing privileges, thus controlling cost by limiting the number of distinct components that must be assured. [9, 41: pp. 1281-1282, 30: p. 7]

4.1.9. Authorizing Operations entailing *programmability*.

*… **programmability** is to distinguish that which is the option from that which is the rule such that the range of options complement the rule. The model designer's tasks are to formalize a language of versatility, to expose the versatility of, to control the exposure of that versatility, to regularize the alternatives of, abstract the interface of, to delay the binding of, to extend an interface's representation beyond a "binary switch" toward a "conversational dialogue." And in so doing the options become extensions of the rule that give flexibility to the stakeholders' application of system features and to their perception of the "problem solving" tasks using it.* [50: p. 35, 9]

Managing authorizations and security protocols requires flexible and convenient tools to accommodate prompt response to policy and environmental change [34]. Change arises as the natural expansion and contraction of authority or protections. It may require restructuring domains to redistribute responsibility, to compensate for evolving stakeholder priorities, or to respond to some emerging threat. [41: p. 1291, 10: p. 42, 30: p. 9]

4.1.10. Manageable Access entailing *user friendliness*.

*… **user friendliness** results when the system's features accommodate the stakeholders' intention. The model designer's tasks are to promote comfort, to foster self-evidence, to facilitate recognition, to promote explicit consistency, to distinguish among differences, to represent the familiar, to satisfy the observer, to reinforce the connectedness of observer and the system and to reinforce the observer's sense of the system's conformance with his or her belief. And in so doing the model choice appears to the stakeholder as convenient and completely expected.* [50: p. 34, 45]

Coherent management of security protocols involves human-system interaction to establish, review and modify access permissions and monitor system behavior by scanning for violations and emerging threats. The number of users, agents and resources can make managing the security specifications complex, particularly in the presence of networked and cloud-based resources. The security controls for both users and administrators need to be user-friendly with intuitive interfaces that minimize confusion and the effort required – else they deter security diligence. [17, 22, 41: p.1282, 30: p.10]

4.1.11. Complete Mediation entailing *reliability*.

*… **reliability** results from an economy of model features limited to the stakeholders' intentions devoid of extraneous embellishments. The model designer's tasks are to regularize, to bring into conformance, to align with stakeholder rules, to represent as aligned with rules, to promote predictability, to render explainable, to conform to the expected and to eschew the unexpected. And in so doing the choice represents "the truth, the whole truth and nothing but the truth." In the truest sense reliability means that "you get what you bargain for" without unexpected complications or entanglements.* [50: p. 35, 37]

Complete mediation is the consistent monitoring of every attempt to access a protected resource – interdicting those without authorization and virtualizing those that require simulation (e.g. in the cloud users access virtual machines, virtual storage, and even virtual channels, rather than physical resources) [21]. However, the granularity at which mediation applies (user vs. role, resource vs. domain) and is assured is a design choice. Granularity impacts system performance and stakeholder convenience. Complete mediation can be likened to the need for situational awareness, monitoring the surrounding environment for dangers [3]. Unfortunately there is no protection against the undetectable. An unmediated access cannot be vetted and probably cannot be audited – it is effectively invisible. [41: pp.1282-1283]

4.1.12. Assurance entailing *correctness*.

*… **correctness** results from relevant, complete, clear and concise representation of the stakeholders' intentions in model features. The model designer's tasks are to represent stakeholder intentions faithfully, to reflect them consistently, to eschew contradictions, to reflect expectations, to conform to beliefs, to satisfy the stakeholders' intentions and to effectively model concerns. And in so doing the choice reflects "truthfully" the stakeholders' intentions and forms a solid foundation upon which subsequent elaboration may proceed with fidelity.* [50: p. 31, 38]

Recall from Section 1 that "Security assurance is the confidence that an entity meets its security requirements, based on specific evidence provided by the application of assurance techniques." [10: p. 481] Security mechanism design must include a means to validate the mechanism's performance and to ensure that it satisfies the security policy's intent. The mechanism must operate correctly and facilitate validation techniques. [10: p. 478, 30: p. 10, 44]

4.1.13. Auditing entailing *transparency*.

*… **transparency** exposes the intention rendered in a model clearly and thus eschews obfuscation. The designer's tasks are to reveal, to render visible, to portray as to interpret, to disclose, to shed light upon, to unfold, to uncover, to lay bare, to bear witness to the stakeholders' intentions, to be true to those intentions, to make self-evident, to make self-explanatory, to publish and to promote the underlying intentions. And to this end the model designer avoids obscuring stakeholder intentions in either the elaboration of a choice or the application of extensions to it.* [50: p. 32, 26]

Theoretical artifacts may be perfect, but human-made artifacts are subject to failure. A security model must anticipate successful attacks due to any of several possible system failures. These attacks and failures in turn may require new policies to be created and existing policies to be reviewed and changed. Auditing is a security function that detects security failures and enables forensics to diagnose and analyze violations [29]. Auditing is thus integral to effectively managing risk in the present, and in the future. [30: p. 9, 41: p. 1282, 10: pp. 706-708]

4.1.14. Risk Management entailing *extensibility*.

*… **extensibility** results in model features so crafted that extended functionality or additional features may be added with a minimum of cost or disruption to the whole. The model designer's tasks are to expose the "common denominators" of feature functionality, to sharpen the "articulation points" that accentuate the "creases" in the unfolding structure / behavior of the model and to expose the potential for "partnering" among the model choices. And in so doing the choice represents not only the achievement of purpose in the present, but is poised to achieve an evolving purpose in the future.* [50: p. 30, 49]

Security management is a dynamic process of risk identification, assessment and response. Both policies and mechanisms must develop organically – building upon trusted architecture to address emerging threats and evolving stakeholder intentions. Surveying for vulnerabilities, risks and threats is an ongoing task which helps assure security model effectiveness and efficiency. Risk awareness informs the overall intentions of the stakeholders and the cost/benefit analysis that guides an organization's security investment strategy. [30: p. 6, 10: p. 17, 48]

4.1.15. Elegance.

*… **elegance** results from coordinating choices to produce an arrangement where each mutually reinforces the other as they fuse into the unifying intention of the whole. The designer's tasks are to harmonize, to orchestrate, to make whole, to complete, to render the system acceptable by validating the stakeholders' intentions through the system's features. And in so doing the choice resonates with the stakeholders' conception and expectation of its place and role in the whole which satisfies their needs.* [50: p. 36, 39]

An elegant security model would make its constituent threats, policies, mechanisms, resources, users, roles, and assurance more readily apparent to stakeholders. The model would be easier to validate, as elegance makes it more likely that gaps in resonance with stakeholder expectations would be identified. By eliminating unnecessary complexity, elegant models also are more readily adapted to evolving applications and changes in technology. The resulting unified whole will reduce costs and make the model more likely to be understood as trustworthy.

4.2. Reconciling IS security and trust

If an interpretation of security is premised only on mechanisms, assurance can produce only a binary result: The mechanism is secure (i.e. always works) or it is not. A vision of security such as this may have been workable pre-Internet or pre-cloud where the artifact was the preeminent focus and an exhaustive testing discipline may have indeed exercised all possible behaviors of the algorithm, mechanism, or computer program. Today's interconnected systems live in an organic environment where the changing agents and multiplicity of connections realize an almost infinite number of scenarios. A conception of IS design quality must admit to an "ecology" continuously creating and interconnecting organic elements where exhaustive testing is not possible. IS security designers must concede that no artifact is perfectly "secure" unless it is inert; and if it is inert, how is it useful? Choosing to pursue a design theory of security in terms of trust is our assertion that human craftsmanship will always be essential to design – incorporating community, culture, perception, intention and satisfaction as design quality determinants [43].

5. Summary, limitations, and future work

The choice properties and design aspects described in this study are a proposed start to a design theory for trustworthy information systems. TST's choice properties and their application to security design address the need for stakeholders to concentrate on why security requirements exist – what they mean to them. They emphasize how security features enable trust rather than focusing myopically on preventing misbehavior.

The explicit inclusion of both quantitative and qualitative stakeholder concerns offers the prospect of a more comprehensive conception of security than mere mechanisms by recognizing the necessary element of stakeholder expectation and experience in a communal concept of trustworthiness. These stakeholder concerns embody the components of artifact mutability and the principles of form and function identified in the anatomy of a design theory [23].

We have had security design principles for decades [16, 33, 41], but these principles are regularly ignored, at great cost. Extensions to these principles, e.g. [54], have not been widely adopted. There is no unique, optimal set of design guidelines for security, but a sound design theory for information systems security would alleviate many problems, including poor communication among stakeholders and communities. Better-informed design would flow through security design choices to produce more secure and therefore more trustworthy information systems. This trust hinges on recognizing and then reconciling and harmonizing stakeholder intentions. Furthermore, the degree of trust achieved and the cost of doing so can be managed by explicitly managing the portfolio of security design choices described in this study.

We intend this paper as a first step toward a design theory. It is clearly limited by only addressing four of Gregor and Jones' eight components [23]. In addition, there are aspects of security that are not yet incorporated. And we have not tested our framework in practice. However, the security design thinking we have described provides a useful prospectus for gathering perspectives, opinions, and criticism in developing a design theory for security. Our next step is to develop a choice property-guided design methodology – ideally for artifact design and implementation in the field. Although pedagogical applications of TST have shown positive results in improving the design performance among students applying the properties in system modelling, industrial-strength efforts like those expected in design science research are needed [24, 52].

Our contribution to the design of trustworthy information systems is the focus on artifact resonance with stakeholder intentions that squarely addresses the imperative that systems architecture and security design must accommodate and facilitate ongoing adaptation and change. As information systems continue to develop as dynamic, continually evolving agents of stakeholder intentions, the conception of IS security must embrace a broader, more ecological perspective that recognizes information systems as extensions of the human condition.

6. Acknowledgements

We are grateful to the students who have inspired and advanced the development of TST and its application to system modeling. We thank the referees for their careful reading and thorough, insightful guidance in the review process.

7. References

[1] Alexander C., Ishikawa S., Silverstein M., Jacobson M., Fiksdahl-King I., & Angel S. *A Pattern Language*, Oxford University Press, New York, NY, 1977.

[2] Alexander C. *The Nature of Order An Essay on the Art of Building and the Nature of the Universe:* Book I - The Phenomenon of Life, The Center for Environmental Structure, Berkeley, CA, 2002.

[3] Anderson, J. P. "Computer Security Technology Planning Study," ESD-TR-73-51, US Air Force Electronic Systems Division, Bedford, MA, 1972.

[4] Anderson, R. "Why Information Security is Hard – An Economic Perspective," Proceedings of *17th Computer Security Applications Conference*, 2001, pp. 358-365.

[5] Baldwin, C. Y., & Clark, K. B. *Design Rules, Volume 1: The Power of Modularity*, The MIT Press, Cambridge, MA, 2000.

[6] Bell, D. E., & LaPadula, L. J. "Secure Computer Systems: Mathematical Foundations," Technical Report Mitre-2547, Vol. 1, Bedford, MA, USA, 1973.

[7] Ben-Natan, R. *Implementing Database Security and Auditing*, Elsevier Digital Press, Oxford, UK, 2005.

[8] Best, B., Jürjens, J., & Nuseibeh, B. "Model-Based Security Engineering of Distributed Information Systems Using UMLsec," *Proceedings of ICSE*, 2007, pp. 581-590.

[9] Birrell, N. D., & Ould, M. A., *A Practical Handbook for Software Development*, Cambridge University Press, Cambridge, UK, 1988.

[10] Bishop, M. *Computer Security: Art and Science*, Addison-Wesley, Boston, MA, 2003.

[11] Boettcher, C., DeLong, R., Rushby, J. M., & Sifre, W. "The MILS Component Integration Approach to Secure Information Sharing," Proceedings of *IEEE/AIAA 27th Digital Avionics Systems Conference*, 2008, pp. 1.C.2-1 - 1.C.2-14.

[12] Bowen, P., Hash, J, & Wilson, M. *Information Security Handbook: A Guide for Managers*, NIST Special Publication 800-100, 2006.

[13] Brooks F. P. "No Silver Bullet: Essence and Accidents of Software Engineering," *Computer*, 20(4), 1987, pp. 9-19.

[14] Chin, S-K., & Older, S. *Access Control, Security, and Trust*, Chapman and Hall/CRC, London, UK, 2010.

[15] CSA. "Security Guidance for Critical Areas of Focus in Cloud Computing V3.0," Cloud Security Alliance (CSA), cloudsecurityalliance.org/guidance/csaguide.v3.0.pdf, 2011, retrieved 22 August 2013.

[16] Denning, D. E. "A Lattice Model of Secure Information Flow," *Communications of the ACM*, 19(5), 1976, pp. 236-243.

[17] Denning, P. J. "Third Generation Computer Systems," *ACM Computing Surveys*, 3(4), 1971, pp. 175-216.

[18] Ellis, J., & Speed, T. *The Internet Security Guidebook: From Planning to Deployment*, Academic Press, San Diego, CA, 2001.

[19] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Reading, MA, 1995.

[20] Garfinkel, S. with Spafford, G. *Web Security, Privacy and Commerce*, 2nd ed., O'Reilly Media, Sebastopol, CA, 2002.

[21] Garfinkel, T., & Rosenblum, M. "When Virtual is Harder than Real: Security Challenges in Virtual Machine Based Computing Environments," in *10th Workshop on Hot Topics in Operating Systems*, 2005.

[22] Graham, G. S., & Denning, P. J. "Protection – Principles and Practice," AFIPS Proceedings of *Spring Joint Computer Conference*, 1972.

[23] Gregor, S., & Jones, D. "The Anatomy of a Design Theory," *Journal of the Association for Information Systems*, 8(5), 2007, pp. 312-335.

[24] Hevner, A., & Chatterjee, S. *Design Research in Information Systems: Theory and Practice*, Springer, New York, NY, 2010.

[25] ISO/IEC. "ISO/IEC 27000:2012 2ed," ISO/IEC, Geneva, Switzerland, www.ISO.org, retrieved 10 June 2013.

[26] Kaisler, S. H. *Software Paradigms*, Wiley-Interscience, Hoboken, NJ, 2005.

[27] Khoshafian, S. N., & Copeland, G. P. "Object Identity," Proceedings of *ACM Conference on Object Oriented Programming Systems Languages and Applications*, Portland, OR, November 1986, 406-416.

[28] Lakoff G., & Johnson M. *Metaphors We Live By*, University of Chicago Press, Chicago, IL, 1980.

[29] Lampson, B. W. "Computer Security in the Real World," *Computer*, 37(6), 2004, pp. 37-46.

[30] Landwehr, C. E. "Computer security," *International Journal on Information Security,* 1(1), 2001, pp. 3-13.

[31] Loscocco, P., & Smalley, S. "Integrating Flexible Support for Security Policies into the Linux Operating System," Proceedings of *USENIX Annual Technical Conference* (FREENIX Track), 2001.

[32] Meyer, B. *Object-oriented Software Construction*, Prentice Hall, New York, NY, 1988.

[33] McCumber, J. R. "Information Systems Security: A Comprehensive Model," Proceedings of *14th National Computer Security Conference* (Annex to NSTISSI No. 4011), 1991.

[34] Neuman, B. C. "Proxy-based Authorization and Accounting for Distributed Systems," Proceedings of *13th International Conference on Distributed Computing Systems*, 1993, pp. 283-291.

[35] National Security Agency. *Defense in Depth*, Fort Meade, MD, retrieved 23 August 2013, www.nsa.gov/ia/_files/support/defenseindepth.pdf.

[36] OECD. "Guidelines for the Security of Information Systems and Networks: Towards a Culture of Security," OECD, 2002, www.oecd.org/sti/ieconomy/1946962.doc, retrieved 20 August 2013.

[37] Pham, H. *Software Reliability*, Springer, Berlin, Germany, 2000.

[38] Pollack, S. (Ed.). *Studies in Computer Science*, Mathematical Association of America, Washington, DC, 1982.

[39] Raymond, E. S. *The New Hacker's Dictionary,* 3rd ed., The MIT Press, Cambridge, MA, 1996.

[40] Rushby, J. M. "Design and Verification of Secure Systems," Proceedings of *8th ACM Symposium on Operating Systems Principles*, 1981, pp. 12-21.

[41] Saltzer, J. H., & Schroeder, M. D. "The Protection of Information in Computer Systems," *Proceedings of the IEEE*, 63(9), 1975, pp. 1278-1308.

[42] Sandhu, R., Coyne, E. J., Feinstein, H. L., & Youman, C.E. "Role-Based Access Control Models," *Computer*, 29(2), 1996, pp. 38-47.

[43] Schon, D. A. *The Reflective Practitioner, How Professionals Think in Action*, Basic Books, New York, NY, 1983.

[44] Schneider, F. B. "Enforceable Security Policies," *ACM Transactions on Information and System Security*, 3(1), 2000, pp. 30-50.

[45] Shneiderman, B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction,* 2nd ed., Addison-Wesley, Reading, MA, 1992.

[46] Scott, M. L. *Programming Language Pragmatics,* 2nd ed., Morgan Kaufmann, Maryland Heights, MO, 2006.

[47] Stallings, W. *Cryptography and Network Security: Principles and Practice*, 6th ed., Prentice Hall, Upper Saddle River, NJ, 2013.

[48] Stoneburner, G., Goguen, A., & Feringa, A. *Risk Management Guide for Information Technology Systems*, NIST Special Publication 800-30, 2002.

[49] van Vliet, H. *Software Engineering: Principles and Practice*, 3rd ed., Wiley, Hoboken, NJ, 2008.

[50] Waguespack, L. J. *Thriving Systems Theory and Metaphor-Driven Modeling*, Springer, London, UK, 2010.

[51] Waguespack, L. J., & Schiano, W. T. "Thriving Systems Theory: An Emergent Information Systems Design Theory," in *46th Hawaii International Conference on Systems Sciences*, 2013, pp. 3757-3766.

[52] Walls, J. G., Widmeyer, G. R., & El Sawy, O. A. "Building an Information System Design Theory for Vigilant EIS," *Information Systems Research*, 3(1), 1992, pp. 36-59.

[53] Windley, P. *Digital Identity*, O'Reilly Media, Sebastopol, CA, 2005.

[54] Wood, C. C. "Principles of Secure Information Systems Design," *Computers & Security*, 9(1), 1990, pp. 13-24.

[55] Zuse, H. *A Framework of Software Measurement,* Walter de Gruyter, Berlin, Germany, 1997.