A Holistic Security Architecture for Distributed Information Systems - A Categorical Approach

Dimitrios Sisiaridis
University of Northumbria
Newcastle, UK, NE1 ST
d.sisiaridis@unn.ac.uk

Nick Rossiter
University of Northumbria
Newcastle, UK, NE1 ST
nick.rossiter@unn.ac.uk
http://computing.unn.ac.uk/staff/CGNR1/

Michael Heather
Ambrose Solicitors St Bede's Chambers
Jarrow NE32 5JB
United Kingdom
michael.heather@cantab.net

Abstract

Distributed information systems (DIS) are free and open systems, characterized by their non-locality. Security for DIS, is a higher order activity, related to issues as data integrity and interoperability among complex heterogeneous systems. This proposed holistic security approach requires category theory. Security entities and distributed activities e.g. distributed transactions, in a DIS, are expressed as Cartesian Closed Categories and adjoint functors between them, following a four-level modular approach.

1 Security in DIS

Security is increasingly important in modern distributed information systems; they are exposed to a growing number and a wider variety of threats and vulnerabilities. It is related to *interoperability* and issues of *integrity* [Shuey and Spooner, 1997]. Generally, it can be achieved by securing the processes and the channels used for their interactions as well as by protecting the resources against unauthorized access [Doughty, 2003].

Information security requirements correspond to specific security policies; each security policy is materialized through a specific security service, which in turn is implemented using one or more security mechanisms as countermeasures against specific security attacks (Figure 1).

2 Interoperability Issues

In today's global environments, which are based on non-local activities as in distributed information systems, interoperable systems are *free* and *open* [Rossiter and Heather, 2006]. Interoperability itself is a global requirement. In the context of information systems, it is concerned with the inter-communication of data at different and therefore usually heterogeneous localities.

In modern heterogeneous interoperable systems, higher-order operations are needed, as the same conditions applied in different systems may lead to unpredictable results. Interoperability as it is higher-order cannot be handled in a complete and decidable manner by axiomatic methods such as first order predicate calculus [Gödel, 1930].

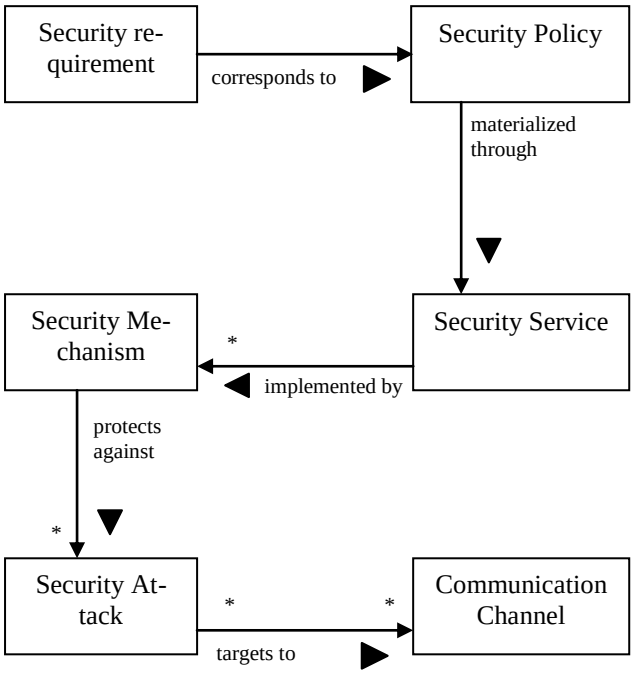

Fig 1: Security in distributed information system

3 The Problem

Current security approaches are characterized by their locality. They can be seen as first-order activities. Bottom-up approaches, such as risk analysis [Coles and Moulton, 2003;Smith and Eloff, 2002;Reid and Floyd, 2001] are subjective; these are more suited to high-level security risks. On the other hand, top-down approaches (e.g. baseline approaches), such as ISO/IEC 27001:2005 Specification [ISO, 2005b] and the ISO/IEC 17799:2005 Code of Practice [ISO, 2005a], leave the choice of control to the user; they are most appropriate for low-level security risks.

Organizations usually respond to security threats on a piecemeal basis following hardware and software solutions that inevitably leave gaps and generate inconsistencies, which can be exploited by intruders. A

promising solution is to include security considerations as core processes of the DIS itself.

A complete security strategy needs to be layered to deal with issues such as continuity strategies (threat assessment, risk evaluation & control), security policies, incident response plan, host-based & network-based perimeter and/or perimeterless detection, auditing procedures, fault tolerance and recovery strategies, anti-malware control (intrusion detection, router and firewall security, anti-virus control) as well as legal and regulatory compliance [Stallings, 2002;Moitra and Konda, 2004;Hawkins et al., 2000].

Design for security has very special methodological problems. Risk management itself a very inexact science, even though it relies very heavily on mathematical techniques, usually applies statistical or probabilistic expectation at a confidence level based on some model. Such methods are being applied more and more widely in both private and public enterprises. However when the issue is security, reliance on just probability is often inadequate and even at times meaningless. A system is not secure unless impenetrable. Even if some probabilistic failure in security is tolerable, the sources and types of possible breaches still need to be known precisely. Therefore an analysis of any such system needs to be fully formal. The languages of systems analysis do not normally exhibit this feature. The diagram we have given in Figure 1 illustrates this point. Text in boxes like 'requirement', 'policy', 'mechanism', 'service', 'attack' and 'channel', all need formal definition. However the arrows between the boxes are not formal functions. Distributed information systems may well depend on relations that are not many to one. Thus one to many is outside the ordinary definition of a function and causes difficulties even in simple relational databases at the local level.

This is where category theory can be introduced with great success. Category theory provides a language for diagrams that is as formal as an algebraic expression. Because a category is a class consisting of arrows between objects it also provides a much greater power than functions between sets. A further important property of a category is that it is of the nature of a type. Categories can then provide in a natural fashion the concept of different levels which is implied in Figure 1 but which cannot be made explicitly clear because of the limitations of the language used there. A set theoretic approach is basically a flat one where to express types and levels soon becomes unnecessarily complicated.

4 The Proposed Holistic Approach

A holistic approach with natural closure seems necessary to describe a complete and global view. It embraces all these aspects of security, including systems architecture, policies, procedures and user education providing natural closure with a very high degree of certainty based on the CIA security principles [FIPS, 2003] (namely *confidentiality*, *integrity* and *availability*). It focuses on securing the infrastructure itself by forcing users to adopt best security practices while ensuring that the network is '*secure by design'* rather than by post-rational customization. Nevertheless, it is crucial that any solution must remain simple to implement as well as simple to use from an end-user perspective.

4.1 Category Theory and o-o Paradigm

In the context of DIS, *components* extend the object-oriented paradigm by enabling objects to manage the interfaces they present and discover those presented by others. The object-oriented approach needs to be founded in applied category theory to be complete and decidable [Barr and Wells, 1999]. Category theory provides a formal approach to process simply by the use of the arrow. It is inherently holistic and with intrinsic natural closure. Composability is a cornerstone of category theory [Asperti and Longo, 1991;Mac Lane, 1998]. Fundamental category theory suggests that for physical existence the real world operates as a Cartesian Closed Category. It can be shown [Rossiter et al., 2007] that any realizable system can be conceptually expressed using four interchangeable levels in categorical terms. The implicated categories are Cartesian Closed Categories (CCC) (Figure 2).

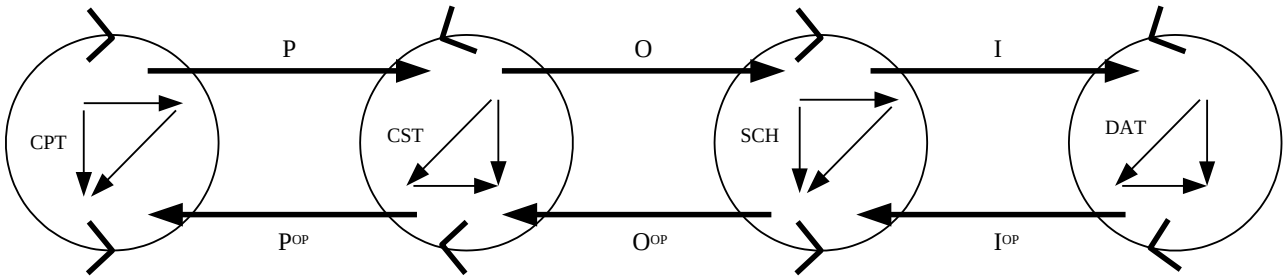

Fig 2: Natural composition of adjoint functors

For matching across the levels in a contravariant manner, the intension e.g. **SCH** is defined with arrows of the form $name \rightarrow type$, while the extension e.g. **DAT** with arrows of the form $value \rightarrow name$. The four levels can be seen as two intension-extension pairs in Figure 3 (**CPT/CST** & **SCH/DAT**), that is concepts/constructs and schema/data respectively.

Before embarking on a full formal description of the architecture, some understanding and informal insight into its interpretation might be useful [Heather and Rossiter, 2001]. The architecture is constructed on four levels. Each type level taken with its adjacent type level acts as a level pair so that there are three level pairs across the four levels. This means that each point at each level is directly related to a point at the other level in the level pair.

At the top level concepts relating to policy and philosophy are defined. For example, object-oriented abstractions are to be declared at this level. In principle, only one instance need be defined here. In a coherent system there can be only one collection of such types. With the open-ended nature of object-oriented structures, however some extensibility may be required.

At the second level schema construction facilities are defined. Each system will have its own type definition. For example constructions would include record-types as an aggregation of single- or multi-valued data field-types while relations would include table-types as an aggregation of single-valued data fields.

At the third level the schema for each application is defined. There will clearly be many intensions defined in an organization, one for each application. Typing of names and other constraints will be applied to data objects and their methods.

At the fourth level the data values for each application are defined. There will be one collection of data values for each schema, the values being consistent with the types of names and constraints of the schema. Data values may be simple objects as in relations or complex objects as in computer-aided design and multimedia systems.

Between each level the mappings are strictly defined by their starting and terminating points in the respective level types. We look first at the mappings in the downward direction (decreasing abstraction). Between levels 1 and 2, Concepts **CPT** and Constructs **CST**, there is the mapping of type Policy P acting as a level pair. Between levels 2 and 3, Constructs **CST** and Schema **SCH**, there is the mapping of type Organization *O* acting as a level pair. Between levels 3 and 4, Schema **SCH** and Data **DAT**, there is the mapping of type *Instance I* acting as a level pair. In the upward direction (increasing abstraction) there are the dual mappings I^{op}, O^{op} and P^{op}.

For the authors, "…*a system is held together by adjointness*…". Interoperability is expressed as the adjointness $\langle F, G, \eta, \varepsilon \rangle$ where $1_L \leq GF$ if and only if $FG \leq 1_R$, as can be seen in Figure 4. *Naturality* is based on the ordering and interoperability of the two free and open represented category systems expressed in the form of CCC (triangles represent unique correlation of components of the system – functors *F* and *G* are the free and underlying functor, respectively). The 4-tuple above has four components: the free functor *F*, the underlying functor *G*, the unit of adjunction η and the counit of adjunction ε.

In more detail adjointness characterizes the unique relationship between Cartesian Closed Categories (that is categories of real-world objects). There is a lower-limit functor (*F*) that preserves co-limits and right-adjoint to (*F*) is an upper-limit functor (*G*) which preserves limits.

Using adjointness between categories **SCH** and **DAT** as examples, the critical comparison is between the arrows f in category type **SCH** and the arrows g in category type **DAT**. It is defining the f in terms of the functors *F* and *G* and the arrow g. We compare an object a with the result of $G \circ Fa$, written simply as GFa, as assigned to category **SCH**. In effect an object in **SCH** is compared with the result obtained by applying functor *F* and then in turn functor *G* to the result. This comparison is a natural transformation η involving type changing: from $a \rightarrow Fa \rightarrow GFa$. This arrow η is called the unit of adjunction. The comparison is made in the context of the corresponding object Gb which maps b in **DAT** to **SCH** so that the condition of adjointness holds, that is $Gg \circ \eta_a = f$.

The perspective of the mapping f can be adjusted to that of the mapping g using the condition that $\varepsilon_b \circ Ff = g$. The arrow ε is the counit of adjunction and a natural transformation comparing FGb to *b*. The view, based on equation solving, is that there is a functorial way to relate any arrow $g : Fa \rightarrow b$ to an arrow $f : a \rightarrow Gb$ in such a way that f solves the equation $g = \varepsilon_b \circ Fy$ and that the solution is unique for either some arrow *y* or object *y* in category **SCH**.

Examples of left adjoints are enrichments such as taking a graph to a category, a set to a group, a set to a preorder and a collection of record keys to hashed addresses. The corresponding right adjoints qualitatively identify the enrichment, ensuring that a number of type restrictions are satisfied.

The notation we use here for an adjunction is as follows. Consider an object a in category **SCH,** an object b in category **DAT** and mappings: $F : \mathbf{SCH} \rightarrow \mathbf{DAT}$, $G : \mathbf{DAT} \rightarrow \mathbf{SCH}$.

Then if there is an adjunction between *F* and *G* (*F*—|*G*), we write the 4-tuple $\langle F, G, \eta_a, \varepsilon_b \rangle : \mathbf{SCH} \rightarrow \mathbf{DAT}$ to indicate the free functor, underlying functor, unit of adjunction and counit of adjunction respectively.

From an application viewpoint, a useful view of an adjunction is that of insertion in a constrained environment. The unit η can be thought of as quantitative creation, the counit ε as qualitative validation. There is then a relationship between the left and right adjoints such that η represents quantitative identification and ε qualitative identification.

In Figure 2 each level pair involves two functors for example *P* and P^{op}, O and O^{op}, I and I^{op}. *I* and I^{op} represent instantiation and naming, *O* and O^{op} organization and meta and *P* and P^{op} policy and metameta, respectively. If the system is coherent adjointness will hold between each pair of functors involved in a level

pair and between any compositions of functors across the level pairs, as shown in Figure 2.

5 Security in Terms of Category Theory

The facets of Figure 1 are formally defined in category theory as follows. A 'requirement' is an underlying functor; 'policy' is a natural transformation; a 'mechanism' is an appropriate functor such as a covariant functor for a '*service*' and a contravariant functor for an '*attack*'; a '*channel*' is some arrow that in category theory needs to be properly defined. The verbs '*corresponds to*', '*materialized through*', '*implemented by*', '*protect against*' and '*targets to*' are all features of adjointness which would allow formal typing of the various components. The archetypal levels of Figure 3 are needed to dimensionalize the full structural form corresponding to the systems language of Figure 1.

Category theory provides higher-order facilities for handling global security. In such a framework, every local solution can be integrated with others, to have the whole view ultimately. Thus, a system can be designed initially in a secure manner following a holistic approach (top-down view), which can be enhanced subsequently by further effective security solutions including those focusing on raising security awareness by minimizing human errors (bottom-up approach).

6 The Proposed Framework

The proposed Holistic Security Framework is developed in two parallel stages. In order to define the high levels of the framework (*stage 1*), objects and object hierarchies (presented in the form of UML diagrams) are categorified into categories. Current analytical techniques used for representing security techniques such as UML and CORBA are expressible over only two or three levels of the architecture shown in Figure 3 so they lack the systemic closure of the categorical approach. But at least we must show that all the basics from the object-oriented paradigm notions, such as inheritance, polymorphism, polyinstatiation, and collections are included in our categorification as all of them are well-expressed using UML.

In *stage 2*, the inner complexity of each of the levels and the mappings between them are described, following the 'process' approach of the distributed system itself [Rossiter et al., 2006]. By following this approach, we must show the event-ordering in local and global conditions in a clear way, as DIS communication including that relevant for security purposes takes place between processes by exchanging messages. Such exchange means that first-order predicate calculus, which serves well for local security problems, is not enough for a systemic approach, in complex environments, as found for example in the Internet.

It is not easy to give an example of a security solution that is relevant to the current ideas. This is because current solutions, top-down and bottom-up, are either platform-based, language-based or service-based. Consequently such solutions are still local and partial.

6.1 The Process Approach

A *distributed computation* M ($M = \{P, W\}$), through its lifetime, is composed of a dynamic group of processes ($P = \{p_1, p_2, ..., p_n\}$) running on different resources and sites expressed in the form of a group of communication channels W. The processes (P) have a disjoint address space and communicate with each other by message passing via W using a variety of mechanisms, including unicast and multicast. While these processes form a single, fully connected logical entity, low-level communication connections (e.g. TCP/IP sockets) may be created and destroyed dynamically during program execution.

Parallel computations that acquire multiple computational resources introduce the need to establish security relationships not simply between a client and a server, but among potentially a hundred or more processes that may span many administrative domains (e.g. computational grids). The communication channels between correct processes are authenticated and protect the integrity and secrecy (privacy). The activity of each sequential process is modeled as executing a sequence of events. A sequence of all the events in a process constitutes a local history. A global history of the computation contains all the events. In an asynchronous system, information may flow from one event to another either because the two events are of the same process and thus may access the same local state, or because the two events are of different processes and they correspond to the exchange of a message [Coulouris et al., 2005]. A binary relation '$\rightarrow$' ("happens before") over the events of the system can be defined in order to express the sequential process of events. Certain events of the global history may be causally unrelated. For two distinct events e and e', neither $e \rightarrow e'$ nor $e' \rightarrow e$ is true. Such events are called 'concurrent', written as $e \| e'$.

Recent work with category theory has shown its potential in relating process order to notional time through the adjointness between monad and comonad categories [Heather et al., 2008].

6.2 Integrating top-down and bottom-up approaches with Category Theory

The holistic security architecture, in categorical terms, can be visualized as mappings between pairs of adjoint functors with abstractions derived from the analysis of the current research. Thus, a process, that is a group process, or a distributed transaction (e.g. a distributed computation as a group of processes, each one consisting of a series of events) can be broken up into a series of composed adjoints to give the power and flexibility of a modular approach. For example, local extensionalities such as local security policies are interconnected one with another through global intentionality

as in a global security policy or meta-policy framework by integrating local slice categories within local policy security domains where each one corresponds to a specific security policy.

Following the bottom-up view, the applied mechanisms and controls can be evaluated against security policies, security services and security controls. In the other direction a parallel top-down view provides an insight of the desirable security level of the system based on the CIA security principles, applied during the design of the system. The integration of the two approaches is the heart of a global, systemic view for handling security in distributed information systems. It provides the flexibility to draw the boundaries of the applied security in the distributed system. At the same time, the system can be re-configured, based on the behavior of the unit and co-unit of each of the adjunctions in the mappings between the levels (Figure 5).

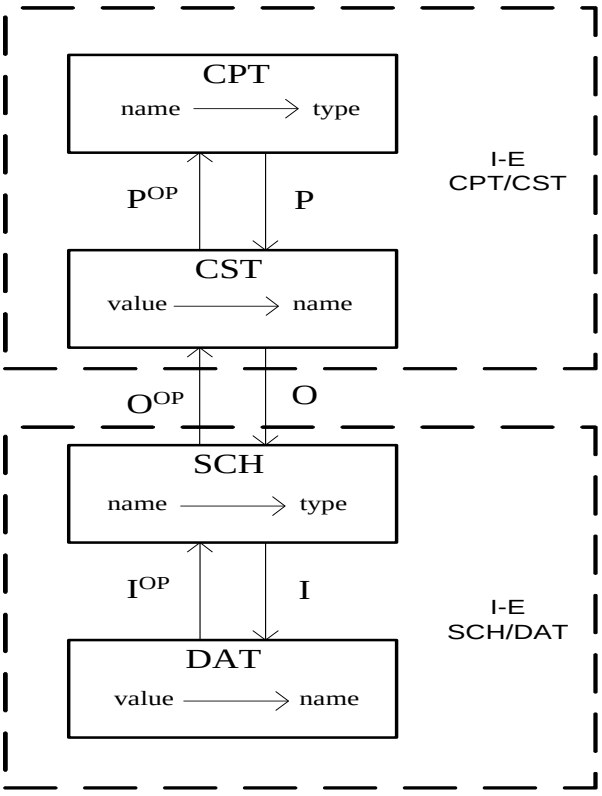

Fig 3: Four levels defined with contravariant functors and intension-extension pairs

7 Conclusions

Current security approaches like baseline approaches, risk management etc. are characterized by their locality. They are based on axiomatic set theory. Security for modern, complex and usually heterogeneous distributed information systems is based on higher order activities.

The results of this current project show that global security for interoperability across heterogeneous distributed information systems can benefit from the use of category theory. A holistic, modular security approach provides natural closure and follows the 'process' approach of the distributed system itself. The proposed framework is in two stages, the first involving categorification of objects and object hierarchies in the form of UML diagrams and the second a detailed investigation of event-ordering in the processes involved.

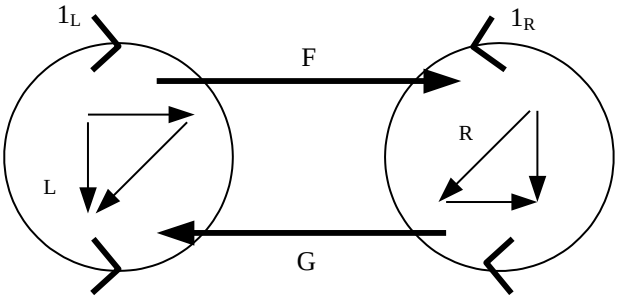

Fig 4: Adjointness between two systems

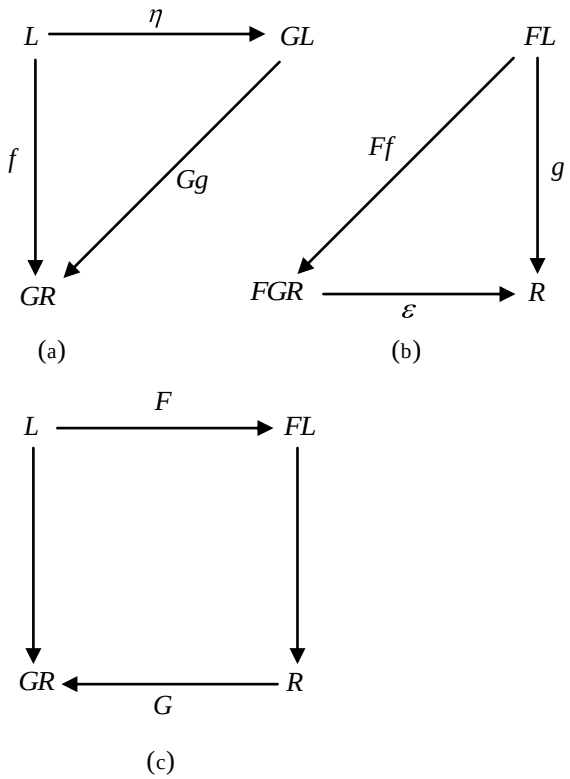

Fig 5: Adjointness between two systems L & R
(a): the unit of the adjunction,
(b) the co-unit of the adjunction,
(c) adjoint functors *F* & *G*

References

[Asperti and Longo, 1991] Asperti, A. & Longo, G. *Categories Types and Structures - An introduction to Category Theory for the working computer scientist*, Foundations of Computing Series, M.I.T. Press, 1991

[Barr and Wells, 1999] Barr, M. & Wells, C. *Category Theory for computing science*, International series in computer science, Montreal, Canada, Les Publications Centre de Recherches Mathematiques, 1999

[Coles and Moulton, 2003] Coles, S. R. & Moulton, R. Operationalizing IT Risk Management. *Computers & Security*, 22 (487-493), 2003.

[Coulouris et al., 2005] Coulouris, G., Dollimore, J. & Kindberg, T. *Distributed Systems: Concepts and Design*, Addison-Wesley/Pearson Education, 2005

[Doughty, 2003] Doughty, K. Implementing enterprise security: a case study. *Information Systems Control Journal*, 2 (99-114), 2003.

[FIPS, 2003] FIPS Standards for security categorization of federal information and information systems. IN U.S Department of Commerce (Ed.), National Institute of Standards and Technology (NIST), Federal Information Processing Standards Publication, 2003.

[Gödel, 1930] Gödel, K. Die Vollständigkeit der Axiome des logischen Funktionenkalküls, *Monatshefte für Mathematik und Physik*, 37 (349-360), 1930. Reprinted in Feferman, S, ed. Gödel Collected Works, volume 1, publications 1929-1936, Oxford, p.102-122 (even numbers) original; p.103-123 (odd numbers) translators Bauer-Mengelberg, Stefan, \& Heijenoort, Jean van (1986).

[Hawkins et al., 2000] Hawkins, S., Yen, C. D. & Chou, C. D. Awareness and challenges of Internet security. *Information Management & Computer Security*, 8 (131-143), 2000.

[Heather and Rossiter, 2001] Heather, M. & Rossiter, N. The Anticipatory and Systemic Adjointness of E-Science Computation on the Grid, in *Proceedings of Computing Anticipatory Systems, CASYS '01*, Liège, Dubois, D. M. (Ed.), 627 (565-574), 2001.

[Heather et al., 2008] Heather, M., Rossiter, N. & Sisiaridis, D. The Semantics of Jitter in Anticipating Time Itself within Nano-Technology, in *Proceedings of Computing Anticipatory Systems, CASYS '07*, Liège, Dubois, D. M. (Ed.), 12pp in press, 2008.

[ISO, 2005a] ISO Information Security Management ISO/IEC 17799:2005 Code of Practice. International Organization for Standardization, 2005.

[ISO, 2005b] ISO Information Security Management ISO/IEC 27001:2005 Specification. International Organization for Standardization, 2005.

[Mac Lane, 1998] Mac Lane, S. *Categories for the working mathematician*, Graduate Texts in Mathematics, New York, Springer-Verlag, 1998

[Moitra and Konda, 2004] Moitra, D. S. & Konda, L. S. An empirical investigation of network attacks on computer systems. *Computers & Security*, 23 (43-51), 2004.

[Reid and Floyd, 2001] Reid, C. R. & Floyd, A. S. Extending the risk analysis model to include market-insurance. *Computers & Security*, 20 (331-339), 2001.

[Rossiter and Heather, 2006] Rossiter, N. & Heather, M. Free and Open Systems Theory, in *Proceedings of 18th European Meeting on Cybernetics and Systems Research (EMCSR-2006)*, University of Vienna, Trappl, R. (Ed.), 1 (27-32), 2006.

[Rossiter et al., 2007] Rossiter, N., Heather, M. & Nelson, D. A. A Natural Basis for Interoperability, in: Enterprise Interoperability: New Challenges and Approaches, Doumeingts, G, Műller, J, Morel, G, & Vallespir, B, (edd) (417-426) 2007.

[Rossiter et al., 2006] Rossiter, N., Heather, M. & Sisiaridis, D. Process as a World Transaction, in *Proceedings of Proceedings of ANPA 27*, Cambridge University, (36pp), 2006.

[Shuey and Spooner, 1997] Shuey, L. & Spooner, L. D. *The architecture of Distributed Computer Systems*, Addison-Wesley, 1997

[Smith and Eloff, 2002] Smith, E. & Eloff, H. P. J. A prototype for assessing information technology risks in Health Care. *Computers & Security*, 21 (266-284), 2002.

[Stallings, 2002] Stallings, W. *Cryptography and Network Security: principles and practice*, Prentice Hall, 2002