# The Work of Dean Rosenzweig: A Tribute to a Scientist and an Innovator

Andre Scedrov
University of Pennsylvania
Department of Mathematics
Philadelphia, PA 19104-6395 USA
+1 (215) 898-5983
scedrov@math.upenn.edu

## ABSTRACT
Dean Rosenzweig, who passed away in January 2007, was a distinguished mathematician and computer scientist. We highlight his contributions to modeling, analysis, and testing of network security protocols, and his work on information technology used in the Zagreb Stock Exchange.

## Categories and Subject Descriptors
F.3.1 [**Logics and Meanings of Programs**]: Specifying and Verifying and Reasoning about Programs – *invariants, mechanical verification, specification techniques;* F.1.1 [**Computation by Abstract Devices**]: Models of Computation

## General Terms
Security, Theory, Verification.

## Keywords
Cryptographic Abstract Machine, Cryptographic Protocols, Model-based Testing

## 1. DEAN ROSENZWEIG (1949 – 2007)

Dean Rosenzweig was a distinguished mathematician and computer scientist. A Professor at the University of Zagreb and leader of research groups in theoretical computer science and in logic and foundations of mathematics, he made significant contributions to logic, computer security, and foundations of software engineering. Professor Rosenzweig was also heavily involved in building up the information technology used in the Zagreb Stock Exchange; thus he was not only a great scientist and educator, but he also helped build technology for a new society.

Here we highlight Professor Rosenzweig's contributions to modeling, analysis, and testing of network security protocols [14] – [16], and his work on information technology used in the Zagreb Stock Exchange. His work on program specification and verification, evolving algebras, and abstract state machines [1]-[13] and [17]-[19] is described in eloquent tributes by Egon Börger [20] and Yuri Gurevich [21].


*ESEC-FSE'07*, September 3–7, 2007, Cavtat near Dubrovnik, Croatia.


## 2. CRYPTOGRAPHIC ABSTRACT MACHINE

The Cryptographic Abstract Machine [15] is an executional model of cryptographic actions, independent of the concrete cryptographic procedures employed and independent of the abstraction level of the underlying model of cryptography. This model is motivated both by a theoretical purpose of relating the dynamics of protocol executions at different levels of abstraction and by a practical purpose of enabling automatic generation and testing of provably correct code implementing protocol roles from high level specifications.

Three such levels of abstraction may be discerned in the literature:

1) Formal model (also called Dolev-Yao model), where only the abstract structure of cryptographic messages is represented, typically by terms of a vocabulary, abstracting away from their concrete representation and concrete cryptographic algorithms,

2) Computational model, admitting that messages are bitstrings, operating under complexity-theoretic assumptions on cryptographic algorithms,

3) Programming APIs, relying on concrete message-encoding schemes and services of a concrete cryptographic library and/or device.

The last two levels are usually related by complexity-theoretic conjectures on combinatorial problems underlying cryptographic algorithms, such as factoring or discrete logarithm. The relationship between the first two levels is an active research area, including the work of Abadi and Rogaway, the work of Canetti, the work of Backes, Pfitzmann, and Waidner, and others.

Rosenzweig and Runje [15] proposed the Cryptographic Abstract Machine (CrAM) in the context of a broader program, with the goal of deriving provably secure code implementing cryptographic protocols. They saw the problem of achieving that goal as composed of the following four aims:

1) Develop a language of abstract message patterns, akin to the usual messages-and-arrows representation, supporting formal proofs under the assumptions of abstract cryptography;

2) Decouple the message patterns from the assumptions of abstract cryptography, allowing their direct interpretation also in computational models and/or concrete implementations of cryptographic algorithms;

3) Provide a framework for translating formal proofs into computational settings, under the common assumptions on cryptographic algorithms;

4) Generate provably correct implementations of pattern-based protocol descriptions for a given programming language and a cryptographic programming library.

The first aim was largely accomplished in [14]: a message pattern language was proved there to be universal for analysis and synthesis of cryptographic messages under a model of abstract cryptography. A formal proof establishing impossibility of an attack with message patterns also proves impossibility of an attack in the context of abstract cryptography.

In [15], the third aim was left for future work. The authors' intent was to develop a framework for systematic translation of proofs for the abstract cryptography model into proofs for different computational models, with target theorems of form: abstractly safe protocol + computationally safe algorithms = computationally safe protocol, for different notions of 'computationally safe'. The authors expected that the properties of abstract cryptography used in a formal proof, together with the computational security criterion desired, would largely dictate the computational security assumptions on algorithms needed. The authors saw this intended future work as a direct continuation and application of the work of Abadi and Rogaway, and Micciancio and Warinschi. The CrAM would to be an appropriate setting, since it provides a simple and precise way of saying that cryptographic agents at different abstraction levels `do the same': they execute the same CrAM program in different environments.

The second and the fourth aims were addressed in [15]. The paper first revisited the message pattern language of [14] in order to allow different models of cryptography, at different levels of abstraction, to be plugged into patterns. For the purpose of [15], the authors intentionally disregarded interpretations of cryptographic objects, and placed a minimal set of assumptions on a model of cryptography needed to prove equivalence of message patterns and CrAM programs. The modified language of message patterns recaptured the universality properties of patterns w.r.t. the model of abstract cryptography of [14]. The main result of [15] was a generation of a provably correct implementation of a pattern-based protocol description for a given programming language and a cryptographic programming library. This was accomplished by compiling patterns to CrAM code, in a provably correct and complete way. In view of the representation of protocol roles of [14] as sequences of match-create pairs of patterns, Rosenzweig and Runje obtained the compilation of protocol roles to CrAM code automatically, by pasting and glueing with instructions for input-output.

The intuition of a protocol role as an interactive machine with input and output of messages got a concrete form in the CrAM, which was also decoupled from any specific model of encryption, and worked with any reasonable model supported by the vocabulary and the assumptions.

Rosenzweig and Runje showed that both compilation algorithms were both correct and complete, which means that CrAM programs are at least as strong as patterns, in any specific instance of both.

# 3. MODEL-BASED TESTING OF CRYPTOGRAPHIC PROTOCOLS

Modeling is a popular way of representing the behavior of a system. A very useful type of model in computing is an abstract state machine which describes transitions over first order structures. The general purpose model-based testing tool Spec Explorer (used within Microsoft, also available externally) uses such a model to perform a search that checks that all reachable states of the model are safe, and also to check conformance of an arbitrary .NET implementation to the model. Spec Explorer provides a variety of ways to cut down the state space of the model, for instance by finitizing parameter domains or by providing predicate abstraction. It has already found subtle bugs in production software.

First-order structures and abstract state machines over them are also a useful way to think about cryptographic protocols, since models formulated in these terms arise by natural abstraction from computational cryptography.

Rosenzweig, Runje, and Schulte explained this abstraction process, 'experiments as structures', and argued for its faithfulness in [16]. They showed how the Dolev-Yao intruder model fits into Spec Explorer. The actions of the Dolev-Yao intruder are the 'controllable' actions of the testing framework, whereas the actions of protocol participants are the 'observable' actions of the model. Under this view, the general purpose software testing tool quickly finds known attacks, such as Lowe's attack on the Needham-Schroeder public-key exchange protocol.

A new 'behavioral' theory of algorithms has been developed in recent years in a series of papers by Blass and Gurevich, by Rossman, and by Rosenzweig and Runje [19]. The main point is that algorithms can be mathematically captured at their own native level of abstraction, for instance, the native level of abstraction of the Euclidean algorithm is that of Euclidean rings. Algorithms operate over abstract first-order structures, well-studied and familiar in mathematical logic, algebra and abstract mathematics in general.

The techniques developed for behavioral theory suggest a natural representation of Dolev-Yao assumptions in first-order structures and a natural mapping of ad-hoc notations present in abstract models of cryptography. Unlike the static abstract models, which necessarily invoke additional proof-theoretic devices to capture dynamic aspects, the behavioral theory explicitly targets the dynamic behavior of algorithms semantically. By recent work on

behavioral theory mentioned above, this also includes interactive algorithms talking to an environment between steps, and within a step, thus allowing a direct representation of the abstract content of oracle algorithms and adversary games typical of computational cryptography. In the framework of intra-step interactive algorithms, the exact abstract representations of computational security notions, defined in terms of adversary games, emerge clearly. The experiments of asymptotic computational cryptography can be naturally represented in terms of interactive algorithms over first-order structures. This is the 'experiments-as-structures' paradigm introduced in [16], which provides a setting for soundness and completeness proofs. The abstract content of these proofs is clearly separated from the probabilistic aspects.

## 4. INFORMATION TECHNOLOGY FOR THE ZAGREB STOCK EXCHANGE

During the ten years that he had been associated with the Zagreb Stock Exchange (ZSE), Dean Rosenzweig had a pivotal role in technical development of at that time a young institution. The area that his advice, influence and work covered is wide and truly impressive both in scale as well in scope.

His work ranged from theoretical analysis of characteristics and properties of different trading paradigms, principles and rules, over to architectural design of main exchange trading and information systems, and finally to actual implementation of many key components of trading system MOST and real-time information system MOSTich, which has been in use by the exchange for many years.

The scope of his work on ZSE was as well impressive, from designing secure and authenticated communication protocols, encryption implementation, stream compression, writing the parser and interface repository for CORBA ORB, to finally writing the key server algorithms for order matching and trading.

But even with such impressive list of his work at ZSE, which could easily fit a group of great people and not just a single one, perhaps Dean Rosenzweig's biggest influence was his mentorship, open mindedness, readiness to cope with any problem that appeared, and true friendship.

## 5. REMEMBERING DEAN

Dean's year at Microsoft Research Redmond gave us an opportunity to reconnect after many years. He visited Penn in December 2004 and gave a fascinating seminar on his work, then in progress, on model-based testing. In turn, I spent several wonderful days in Seattle in May 2005. This was the same Dean as I knew him during my high school and undergraduate days in Zagreb: Dean with a contagious zest for life, Dean with a keen mind and an intuition for identifying the real point of the matter at hand and willing to take new paths to get there, Dean who appreciated subtleties and ironies of the human condition, Dean the food and wine connoisseur, Dean fluent in many languages. His work and his contributions are here to stay, but the man, alas, is gone. He is and will be sorely missed. May his memory be eternal.

## 6. ACKNOWLEDGMENTS

I would like to thank Paola Glavan, Yuri Gurevich, Davor Runje, Davorin Rusevljan, and Wolfram Schulte for their kind help with the preparation of this tribute. Davorin Rusevljan contributed Section 4.

## 7. REFERENCES

[1] Börger, E. and Rosenzweig, D. From Prolog Algebras Towards WAM – A Mathematical Study of Implementation. In E. Börger *et al.,* eds., *Computer Science Logic, 4-th Workshop (CSL'90)* (Heidelberg, Germany, October 1-5, 1990). Springer LNCS vol. 533, 1991, 31-66.

[2] Börger, E. and Rosenzweig, D. WAM Algebras – A Mathematical Study of Implementation, Part 2. In A. Voronkov, ed., *Logic Programming, First Russian Conference on Logic Programming* (Irkutsk, Russia, September 14-18, 1990) – *Second Russian Conference on Logic Programming* (St. Petersburg, Russia, September 11-16, 1991). Springer LNCS vol. 592, 1992, 35-54.

[3] Glavan, P. and Rosenzweig, D. Communicating Evolving Algebras. In E. Börger *et al.,* eds., *Computer Science Logic, 6-th Workshop (CSL'92)* (San Miniato, Italy, September 28-October 2, 1992). Springer LNCS vol. 702, 1993, 182-215.

[4] Börger, E. and Rosenzweig, D. The Mathematics of Set Predicates in Prolog. In G. Gottlob *et al.,* eds., *Computational Logic and Proof Theory, Third Kurt Gödel Colloquium (KGC'93)* (Brno, Czech Republic, August 24-27, 1993). Springer LNCS vol. 713, 1993, 1-13.

[5] Börger, E. and Rosenzweig, D. Full Prolog in a Nutshell. In D.S. Warren, ed., *Logic Programming, Proceedings of the Tenth International Conference on Logic Programming (ICLP'93)* (Budapest, Hungary, June 21-25, 1993). MIT Press, 1993, 832.

[6] Börger, E., Durdanovic, I., and Rosenzweig, D. Occam: Specification and Compiler Correctness - Part I: The Primary Model. In E.-R. Olderog, ed., *Programming Concepts, Methods and Calculi, Proceedings of the IFIP TC2/WG2.1/WG2.2/WG2.3 Working Conference on Programming Concepts, Methods and Calculi (PROCOMET '94)* (San Miniato, Italy, June 6-10, 1994). IFIP Transactions A-56, North-Holland, 1994, 489-508.

[7] Rosenzweig, D. Distributed Computations: Evolving Algebra Approach. In B. Pehrson and I. Simon, eds., *Technology and Foundations - Information Processing '94, Volume 1, Proceedings of the IFIP 13-th World Computer Congress* (Hamburg, Germany, August 28- September 2, 1994). North-Holland, 1994, 440-441.

[8] Glavan, P. and Rosenzweig, D. Evolving Algebra Model of Programming Language Semantics. In B. Pehrson and I. Simon, eds., *Technology and Foundations - Information Processing '94, Volume 1, Proceedings of the IFIP 13-th World Computer Congress* (Hamburg, Germany, August 28-September 2, 1994). North-Holland, 1994, 416-422.

[9] Börger, E., Del Castillo, G., Glavan, P., and Rosenzweig, D. Towards a Mathematical Specification of the APE100 Architecture: The APESE Model. In B. Pehrson and I.

Simon, eds., *Technology and Foundations - Information Processing '94, Volume 1, Proceedings of the IFIP 13-th World Computer Congress* (Hamburg, Germany, August 28-September 2, 1994). North-Holland, 1994, 396-401.

[10] Börger, E. and Rosenzweig, D. A Mathematical Definition of Full Prolog. *Science of Computer Programming 24, 3* (1995) 249-286.

[11] Börger, E. and Rosenzweig, D. The WAM - Definition and Compiler Correctness. In C. Beierle and L. Plümer, eds., *Logic Programming: Formal Methods and Practical Applications.* Studies in Computer Science and Artificial Intelligence, Elsevier Science B.V./North-Holland, 1995, 20-90.

[12] Börger, E., Gurevich, Y., and Rosenzweig, D. The Bakery Algorithm: Yet Another Specification and Verification. In E. Börger, ed., *Specification and Validation Methods.* Oxford University Press, 1995, 231-243.

[13] Gurevich, Y. and Rosenzweig, D. Partially Ordered Runs: A Case Study. In Y. Gurevich *et al.,* eds., *Abstract State Machines, Theory and Applications, International Workshop (ASM 2000)* (Monte Verità, Switzerland, March 19-24, 2000). Springer LNCS vol. 1912, 2000, 131-150.

[14] Rosenzweig, D., Runje, D., and Slani, N. Privacy, Abstract Encryption and Protocols: An ASM Model - Part I. In E. Börger *et al.,* eds., *Abstract State Machines, Advances in Theory and Practice, 10-th International Workshop (ASM 2003)* (Taormina, Italy, March 3-7, 2003). Springer LNCS vol. 2589, 2003, 372-390.

[15] Rosenzweig, D. and Runje, D. The Cryptographic Abstract Machine. In W. Zimmermann and B. Thalheim, eds., *Abstract State Machines 2004. Advances in Theory and Practice, 11-th International Workshop (ASM 2004)* (Lutherstadt Wittenberg, Germany, May 24-28, 2004). Springer LNCS vol. 3052, 2004, 202-217.

[16] Rosenzweig, D., Runje, D., and Schulte, W. Model-Based Testing of Cryptographic Protocols. In R. De Nicola and D. Sangiorgi, eds., *Trustworthy Global Computing, International Symposium (TGC 2005)* (Edinburgh, UK, April 7-9, 2005). Springer LNCS vol. 3705, 2005, 33-60.

[17] Rosenzweig, D. and Runje, D. Some Things Algorithms Cannot Do. Microsoft Research Technical Report MSR-TR-2005-52, 2005.

[18] Blass, A., Gurevich, Y., Rosenzweig, D., and Rossman, B. Interactive Small-Step Algorithms I: Axiomatization. *Logical Methods in Computer Science,* to appear. A preliminary version appeared as Microsoft Research Technical Report MSR-TR-2006-170, November 2006.

[19] Blass, A., Gurevich, Y., Rosenzweig, D., and Rossman, B. Interactive Small-Step Algorithms II: Abstract State Machines and the Characterization Theorem. *Logical Methods in Computer Science,* to appear. A preliminary version appeared as Microsoft Research Technical Report MSR-TR-2006-171, November 2006.

[20] Börger, E. A Tribute to Dean Rosenzweig. http://www.eecs.umich.edu/gasm/dean.html

[21] Gurevich, Y. Abstract State Machines: Remembering Dean Rosenzweig. http://www.eecs.umich.edu/gasm/dean.html