A Holistic Model of Enterprise Development (I)

Hong Li
Water-Logic, Inc. Phone: 614-296-7644, Fax: 866-206-9568, Email: hong@water-logic.com

Abstract: As more and more management and control functions are integrated into industrial systems, industry experience and academic research under Purdue Enterprise Reference Architecture have found that taking unconditional automation as the ultimate goal is highly problematic. The unusual complexity theory developed by Robert Rosen, which emphasizes on the causal impact of organizations in any system, provides the needed theoretical support to identify, explain and predict the key issues in enterprise development. Common concept of machine models and organic models should be reconsidered accordingly. It is necessary, as well, to rethink the conventionally accepted practice of treating human and organizational development as secondary concerns during engineering and technical endeavors in industry. This is the first one of three articles in a series that reports progress in PERA research.

Keywords: Enterprise Engineering, Systems Engineering, Relational Model, Complexity Theory, Enterprise Reference Architecture, PERA (Purdue Enterprise Reference Architecture), Systems Science

1. Introduction

The early efforts in enterprise integration and enterprise engineering can be traced back to the 1960s. Figure 1 shows a brief summary of this history from early industry automation, which was focused on groups of industrial equipment, to the modern enterprise development, which has evolved into organizational networks. The amount of information to be processed has also been dramatically increased during the same period. However, this industrial evolution has brought forth more than quantitative changes in information processing. Fundamental changes in major scientific methodologies have become necessary because the complexity involved has qualitatively changed. Some scholars have recognized that "organized complexity", as shown in Figure 2, is what enterprise integration and enterprise engineering must address. When enterprises have to develop and implement individualized business strategies in a changing environment, relationships within and between enterprises are not the same as those with machine equipment, and these relationships cannot be reduced into statistic averages of unorganized complexity either.

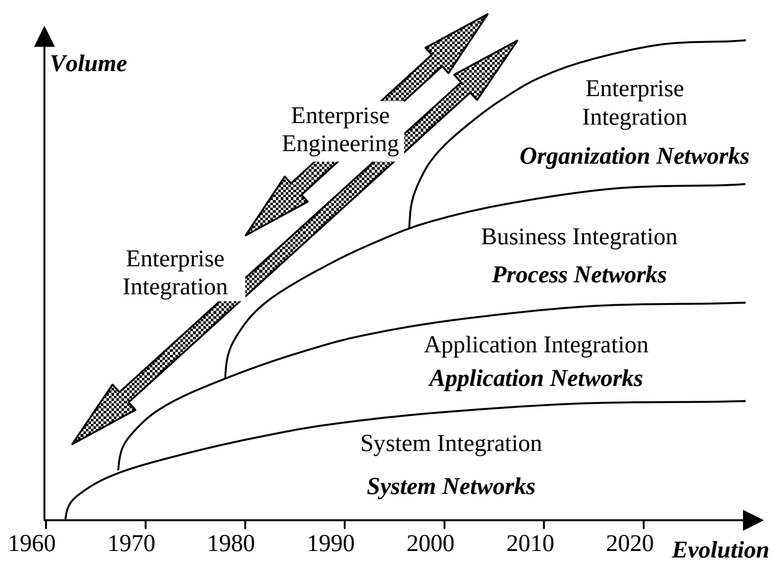


Figure 1. From integration to engineering [Kosanke, 1999]

This series of three articles will demonstrate that, instead of following systems paradigms in Figure 2, many technically oriented research paradigms, including some popular enterprise modeling approaches, still resemble the machine paradigm in Figure 2. Without the necessary holistic perspectives, the intrinsic limits of these technical paradigms can be a strategic error, overlooking the place of the human in enterprise development. The root cause can only be identified by fully recognizing the limitations of the theoretical framework that supports machine development.

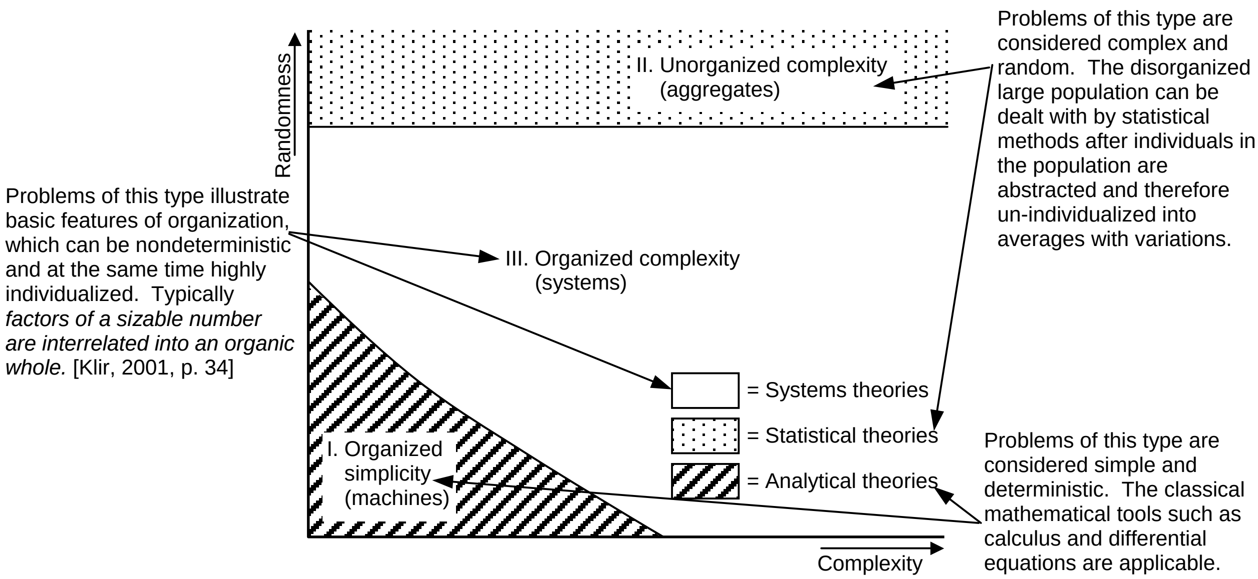


Figure 2. Types of systems with respect to scientific methods [Klir, 2001; Weaver, 1948; and Weinberg, 1975]

With help from Robert Rosen [Rosen, 1978; 1985a; 1985b; 1991; and 2000], the author will present recent progress in the theories and industrial practice of enterprise development. These new theories not only lay out the much needed systems foundation, but also illustrate new and strategic directions for further enterprise development, when two different types of organizational development, organic and mechanical, can be redefined and aggregated accordingly, with the latter as a subset of the former.

These articles will illustrate why with the recognition of human and organizational considerations as well as full life cycle of enterprise development, the Purdue Enterprise Reference Architecture (PERA) has embodied a holistic approach to coping with organized complexity in Figure 2, and what one can learn from modern systems science to benefit studies in enterprise development and beyond, going as far back as the original Aristotelian concepts.

A potential crisis is pending in several related areas: enterprise integration, information systems integration, and systems engineering. Although the computerized approach to full automation at the enterprise level may be set as the ultimate goal, its inherent limitations serve as proof of the existing strategic gap between business needs and academic research. Where the organizational networks, which earmark the needs for modern enterprise development as shown in Figure 1, require the treatment for organized complexity as shown in Figure 2, the currently applied theories for building and operating machines as well as the connections between them will be fundamentally inadequate, if they are expected to provide the candidate solution to completely replace people, or unconditionally substitute electronic connections for human communications, in order for the organizations to maintain sustainable growth under highly competitive and changing environments.

In this article, the author adopts the generic definition of enterprise from [ISO 15704, 2000]: one or more organizations sharing a definite mission, goals, and objectives to offer an output such as a product or service. This term includes related concepts such as extended enterprise or virtual enterprise. Note that by definition an enterprise can also be a project, a program, or an institution [Selznick, 1957]. Even a product can be considered as a specific enterprise entity by this definition [Hong, 2000].

This article[1] is the first one in a series of three that report the progress in PERA research. It will establish a graphical mapping between PERA and Rosen's M-R (Metabolism-Repair [Rosen, 1958, p. 252]) model to identify under Aristotelian-Rosennean causal framework an organic pattern embedded

[1] This article is a revision of one that published on a Web Journal, BioTheory, Inaugural Issue, 2005, available at http://www.rosen-enterprises.com/RobertRosen/HongLiBioTheoryPaper.

within PERA life cycle model. The second article will introduce some mathematical tools of Rosen into the study of enterprise development to further discuss the properties of Rosennean organic model vs. machine model in the context of enterprise development. The last article will investigate the entailment relationships that underpin the intra- and inter-enterprise relations of an enterprise chain [Li, 1995 and 2000] so that the business values of Rosennean organic approach in enterprise development can be appreciated together with presentation of industrial experiences.

2. Brief Review of PERA and Key Issues with Enterprise Engineering

Quite interestingly, the efforts associated with PERA at the former Purdue Laboratory for Applied Industrial Control (PLAIC) can be traced back to early industrial automation that targeted integration between management/control and the industrial systems in question. The evolution of PERA represents an inevitable path of enterprise engineering that eventually has to break out from the boundaries of traditional engineering. The key issues may be understood based on the definition of enterprise engineering or integration [Li, 2004]. Under the traditional approach, engineering systems are usually treated as systems of machines, which consequently can be considered simple systems [Klir, 2001]. It has been a common practice, for example, in systems engineering to apply the same rules, without discrimination, to hardware, software, and humans as equal components present in the systems [Jackson, 2000].

It has taken years of research and industrial practice for the developers of PERA to recognize and understand the crucial role of human and organizations and its consequences for modern enterprise development programs that are beyond the scope of traditional engineering. The main purpose of the traditional practice is focused on developing equipment that follows predefined procedures after external instructions are received.

In the 1980s, the need for integration for industry automation was identified as part of the initiatives of Computer Integrated Manufacturing (CIM). A reference model, the Purdue Reference Model for CIM, as the industrial guidelines was first developed at PLAIC [Williams, 1989].

Several companies immediately recognized the benefits of the Purdue Reference Model. Later in 2000, The Reference Model was recognized as the foundation of the industrial standard series of ISA SP95, which in turn was the foundation of IEC/ISO 62264 in 2002. On the other hand, however, the empirical data from industry with alarmingly high rates of project failure soon showed that the Reference Model, as an industrial standard for management and control systems, was still not sufficient for implementation projects. The standard was originally developed for guiding hardware and software design. Therefore, it paid little attention to the organizational efforts that are necessary to deliver the enterprise systems, and it treated human related functions mainly as external entities outside the enterprise systems [Williams, 1996].

In other words, one of the most important early lessons learned from the development of the Purdue Reference Model is that computer systems alone can never integrate enterprises or industrial systems that include humans and associated organizations where most of the project failures are caused by people or organizational elements. Standards for enterprise development programs to deliver an enterprise are much more complicated than the technical standards for that enterprise. Within the industrial systems being integrated, more and more instructions to production equipment and services become internally generated. As the level of integration increases from machine operations on the shop floor to mid-term and long-term planning, internal decision procedures cannot be decided the same way as those of automated equipment.

Under the leadership of Professor Theodore Williams of PLAIC, the Industry-Purdue Consortium for CIM, joined by ten major industrial companies, realized that the Purdue Reference Model needed major rework to address the problems of enterprise relationships, technology implementation, and human and organizational considerations that had been bypassed by the Purdue Reference Model. That major rework was carried out from 1989 to 1992. The results turned out to be a rewrite and had to be renamed as PERA, and Purdue Methodology. The former is an architectural model of enterprise development presented as a framework of full life cycle, and the latter a step-by-step implementation guide for the enterprise development program in the form of a master plan, which includes an enterprise business plan or strategic plan as necessary [Williams, 1992 and 1996].

PERA can also be considered as the model of the Purdue Methodology. This model embraces the full life cycle of enterprises from their conceptual birth all the way to their final disposal. It mandatorily contains not only the enterprise information systems for management and control, but also the production systems where the customer products and services are generated. In a sense, it offers a comprehensive definition of industry systems, which can be applied as needed in any business according to their specific missions and visions.

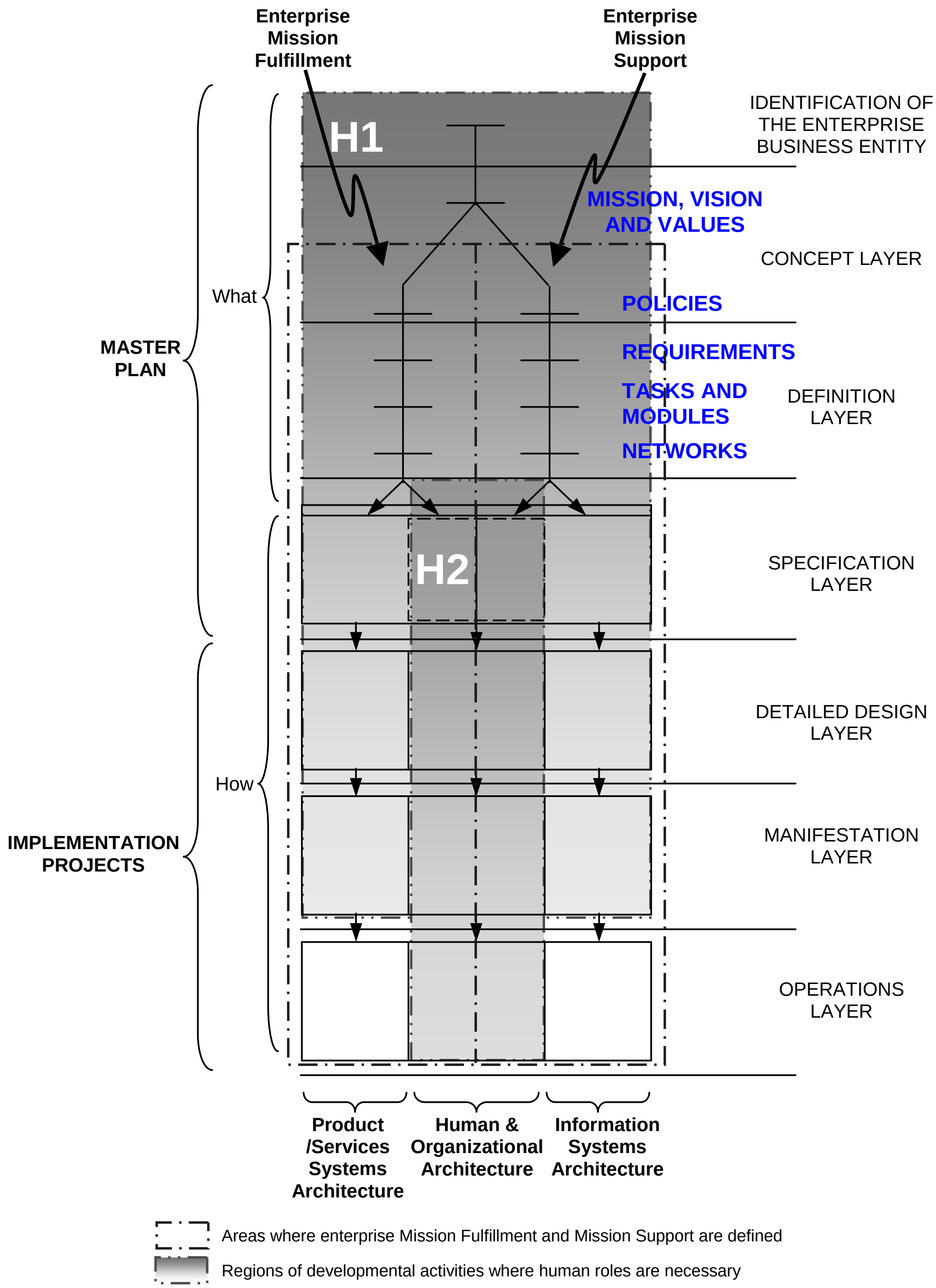


Figure 3. A partial sketch of PERA life cycle model

Figure 3 is a partial sketch of a graphic presentation of PERA. Different from many other enterprise reference architectures of similar types, the life cycle of PERA starts from the very initial formation of the business concepts about the subject enterprise, represented by the phase of Identification of Enterprise Business Entity as shown in Figure 3. That is to say, as indicated by the first three phases from Identification of Enterprise Business Entity to Definition Layer, the enterprise development program as defined by PERA must include the initial business planning processes. Most other current enterprise reference architectures today still treat those business requirements as a given, or *a priori* [Uppington, 1998]. Their life cycles typically therefore start somewhere within the Definition Layer of PERA.

The main focus of many current enterprise reference architectures is to develop technical solutions based on business requirements from external entities. Usually, human roles are either specified as one of the external agents for the reference of man-machine interface design, or specified as one of the internal components of the system together with software or hardware. However, equipped with business requirements that are developed within the life cycle established early, PERA requires, at the Specification Layer in Figure 3, the specifications of the Human and Organizational components of the enterprise be prepared with the highest priority in the requirement development process, in order to establish the comprehensive enterprise requirements. Unlike typical technical standards, the Human and Organizational Architecture of PERA addresses issues from cultural values to political infrastructures, which contain, but are far more than, formal organizational charts or job descriptions.

Therefore, the life cycle model of enterprise development as defined by PERA represents two important aspects of completeness of full life cycles of enterprises:

- A complete life cycle of an enterprise starts with the conception of initial business ideas of the subject enterprise.
- A complete life cycle of an enterprise delivers complete business solutions that should not only include human and organizational solutions together with the technical ones, but also treat the technical ones as dependent on the humans and organizations involved, in terms of the requirement development.

Based on industrial research, PERA fully embraces the complexity involved in an enterprise development program in two dimensions, time- and component-wise. Consequently, the full recognition of the significance of human and organizational elements makes it improper to reduce the whole enterprise system into a computerized information system [Li, 2004]. Inevitably, the life cycle of an enterprise presents a seemingly impossible mission for reductionist-based (see further discussions on reductionist approach in Appendix A) complexity management. As [Weinberg, 1975] pointed out, there is little chance for any Cartesian-based views or state variables to be successful if they are brought into "situations of even modest complexity".

In order to manage the complexity and magnitude of the life cycle so identified, PERA offers a unique simplification package of requirement management that is focused on life cycle partition of the requirement development in order to facilitate the conversion of business requirements into implementation specifications as follows:

- During the development of business requirements, which is represented by the first three phases in Figure 3, no means of physical or electronic implementation, neither machine nor human, should be considered. In the terms of PERA, that region of the enterprise life cycle in Figure 3 is the area of identified requirements about "What" to do, not "How" to do it. The considerations of "How" should start only after "What" has been decided.
- Within this "What" area of PERA, the overall business functions of an enterprise should be divided into two sets, Mission Fulfillment and Mission Support as shown in Figure 3. The former represents the business model that delivers the customer products and services in business terms. The latter represents the business model that conducts the management and control function, also in business terms.
- At the beginning of "How", PERA requires that human components that will implement the functions of Mission Fulfillment and Mission Support be identified first before the machine components, including computers and non-computers, are selected. PERA recognizes that the selection of human functions is dependent upon the economic, political, social, and technical factors. Therefore, the major functions of the Specification Layer in Figure 3 are assignments of the enterprise Mission Fulfillment and Support to their implementers to form three implementation architectures, Human and Organizational Architecture, Customer Product/Service Systems Architecture, and Information Systems Architecture. [Williams, 1998]

In addition to the requirement development of an enterprise, PERA acknowledges the important developmental roles of human activities during the life cycle of enterprises. The shaded areas, H1 and H2, in Figure 3 represent the presence of the tasks, identified by PERA, where human roles in the development processes are irreplaceable because of the needed human innovation, judgment, experience, flexibility, and associated rich communication, etc. H1 is the area where initial business requirements are generated throughout all levels of organizations involved. H2 is the area where human and organizational components, Human and Organizational Architecture in PERA terms are developed. The darker the area, the more human roles are needed during the enterprise development.

Among major enterprise reference architectures, the International Task Force on Architectures for Enterprise Integration identified PERA as the most complete one [Bernus, 1996]. Since 1995, PERA[2] has been one of the major contributors to [ISO 15704, 2000]. On the other hand, some scholars [Chalmeta, 2001; Kosanke, 1996; Ortiz, 1999; Patankar, 1995; Rolstadås, 2000; and Vernadat, 1996] have considered the fact that PERA does not use computerized constructs to be a weakness. There have been attempts [Chalmeta, 2001 and Rolstadås, 2000] within the research circle of computerized enterprise modeling to convert PERA into fully computerized systems based on a belief that the processes of enterprise development should one day be automated by information technology alone [Li, 2004].

The author appreciates these efforts to computerize the tasks associated with requirement development of enterprises, which will bring in more tools within the construct-based computer modeling approach. Nevertheless, a research strategy ultimately targeted at total automation of enterprise development is different from the one set up to facilitate the human as an irreplaceable central agent of enterprise development in general. The strategy for total automation will inevitably leave many needs of human agents unattended.

Should total automation be the ultimate goal of the research, the current human roles involved in the enterprise development would become a kind of temporary or makeshift substitute waiting for a better technology to come. The major research should then be directed mainly towards the endeavor that is looking for more advanced technical solutions, including one for enterprise planning advocated by PERA and Purdue Methodology so that the needs for human roles could be further eliminated from both the enterprise operations and the development processes that develop these enterprises.

This pure technical paradigm described above serves as a strange prescription in response to the discovery of PERA research that necessitated the full life cycle of enterprise development where the importance of innovative business ideas and the roles of humans and organizations are recognized. A key question is whether those problems and issues identified during the implementation of industrial automation are just a sign of the lack of more advanced technology. The answer would be yes if our study were confined within the current theories and technologies that have helped human society invent and develop automated machines.

Even more importantly, given the complexity and magnitude of enterprise development, which have been acknowledged by most researchers, the lack of basic study on the foundations of the different directions, or visions of enterprise development [Li, 2004] makes it impossible to fully explain the difference between the two seemingly uncompromisable approaches, technically oriented total automation vs. business and human oriented development. Needless to say, it is impossible, without the support of consistent theories, to predict what will be entailed by the different approaches.

Although the author identified years ago that the Engineering Design Axioms of Professor Suh as a theoretical base for the design of the life cycle partition of PERA, Suh's axioms only provide one aspect of complex management, which has to start with ready user requirements [Suh, 1990 and 1999]. A more comprehensive theoretical model that is able to underpin PERA's approach to the entire life cycle of enterprise development programs including the origin of the user requirements remained largely missing. Such a model should be able to guide the refinement of PERA.

[Goranson, 1992] pointed out that it was the epistemology of enterprise development that defined the nature of the domain: "This philosophy is, by definition, not primarily driven by technical concerns. Rather, business and sociological constraints of information interaction prevail." Unfortunately the author so far has not found, within the research of enterprise development as well as systems engineering in

[2] Interested readers are referred to the following readings to find more information on PERA and the Purdue Methodology as well as field reviews: [Li, 1995 and 2004] and [Williams, 1992, 1996, 1998, and 1999].

general, any other published work that has explored the epistemological paradigms underpinning the major research hypotheses.

For years after the study of Suh's axioms, the author could not find the necessary theoretical help from contemporary science until he found that there were many similar concerns shared between Rosen's study of organic life and PERA's study of enterprise life cycle. This article demonstrates that in the field of enterprise engineering, or even systems engineering, the counterpart of the question "What is Life?" pursued by Rosen is "What is an Enterprise?" Readers will soon see that this article answers these seemingly innocent questions that lead us to understand fundamentally why PERA should have reservations about computer-based modeling tools and why the mechanical mechanism, which is represented by item 1 in Figure 2, should not be blindly extrapolated into the field of organized complexity in Figure 2.

In the theoretical world of machines, only things that are machine-programmable or machine-executable matter the most. Those may not be true any more once we delve deeper into the world of complex systems such as modern enterprises. Although the more complex world may still include machines, it is those nature-made beings, including the creator of machines, the human beings, that through our own innate non-programmability [Simon, 1977] ultimately make the theories of machines inadequate. This inadequacy is revealed when it is necessary to identify the very reason why PERA has to have its unique definition of full life cycles. The inadequacy is also typified when some popular theories of reductionism cannot differentiate the roles between human and machine.

3. Preparations for Relational Theories of Enterprise Modeling

Many current theories, including those related with the studies of enterprise development, are disqualified for the study of life cycles defined by PERA. One of the important reasons is that they do not explore the purpose, or the needs of their subjects, which have become an inseparable part of PERA life cycle. Many theoretical frameworks are mainly focused on the studies of properties of static structures or dynamic behaviors of subject enterprises without explicit and direct questioning of the underlying purpose. The responsibility to *relate* such a subject enterprise to its business purpose is usually outside these frameworks until the purpose is externally formulated into well-defined business requirements.

When Rosen introduced the four causes of Aristotle into his early study of life [Rosen, 1985b] so that he could reveal the entire entailment structure including the purpose, he, as a theoretical biologist, developed the missing theoretical foundation not only for the study of biological life but also for enterprise development as needed by PERA. As he has demonstrated in his mathematical form, Aristotelian causality addresses *the relations with finality* that only exists within complex systems [von Bertalanffy, 1969]. As a recognized systems scientist, Rosen knew as well that his work established an effective approach to generic organizations in their entirety as needed by systems study [Rosen, 1985b and 1991].

However, in order to establish the connection between Rosen's system and PERA in the field of enterprise development, there is still a need for one more step to complete the extension from the general framework of the four causes to the tangible and complex cases in enterprise development, which include people and organizations as defined by PERA. Only after this validating discussion is finished, can we then apply Rosen's modeling theory with confidence.

Please note that while it is necessary to first establish the needed link between Aristotelian causality with the pragmatic concerns in enterprise development, the Aristotelian-Rosennean causal framework represents a rather unique way of thinking based on Rosennean relational synthesis. There is no ideal approach to translating this all-inclusive relational framework (see Table 1 for the definitions of these Aristotelian causalities) into regular terms of traditional engineering practice, where many people may not be familiar with either the Aristotelian causalities, or the Rosennean complexity[3] theories of relations. In order to preserve the richness of this relational framework, the author prefers keeping the original Greek-English names of Aristotelian causes to possible oversimplification that can be even more misleading.

The Greek-English translation listed in Table 1 may have made the names of the four causes somewhat peculiar. For example, the efficient cause may have little to do with efficiency, and the formal cause does not have much to do with formality either. The readers are however advised not to jump into a conclusion that the efficient cause can be equal unconditionally to an implementer, or the formal cause must have

[3] Although Rosen has given his definition of complexity, this article only uses this concept loosely as a common sense, since it will not change the discussions.

always been a step-by-step plan. The examples given in Table 1 are straightforward in order to prepare the introductory presentation in this article. More detailed discussions in this series of three articles will reveal richer entailment perspectives of the Aristotelian-Rosennean framework, especially when it is applied to organic development where each of the four causes may become relational and contextual. It will turn out that these simplified conclusions may only be applied to the machine model, which is a subset of the organic model.

This relational framework therefore will be presented through a series of examples and cases closely associated with the discussions of the model development, just as peeling off "the layers of its onion skins" to reveal the unusual and rich perspectives underpinning enterprise development. In this way, the readers may hopefully gain new insights into Rosennean relational model by going through the conceptual transitions back and forth between the newly defined organic model and machine model.

In the further discussions, formal expressions based on Rosen's Abstract Block Diagram, a graphic tool for Aristotelian entailment study [Rosen, 1991], will be associated with the semantics in order to help the readers fully understand the details in the context of enterprise development. The reasoning process in this article will rely mainly on the practical rationale of the entailment structure to build the reasoning logic directly based upon common concerns in enterprise development programs. For the main purpose of this article, one should avoid what Weinberg termed "hypermathematisis" [Weinberg, 1975] and stay close to the issues of concern, which otherwise could be easily obscured behind heavy mathematic syntax. Although this may suffer from the lack of a mathematical aura, it is believed that one is still able to maintain the rigorous logic of the inferential process in the thought experiments [Brown, 2000] presented next.

The Appendix B at the end of this article summarizes the graphical syntax of the abstract block diagram, which may help interested readers understand the basic tools for this article. Based on the conceptual model established in this article, the next article in this series will look more into Rosen's mathematical tools, particularly his presentation of category theory, in order to discuss the organic properties that graphical tools are not able to present here.

Figure 4 is such a typical example of a small project: an ancient Greek sculptor working on a piece of stone to produce a sculpture. Then, as through Aristotle's eyes, one could easily identify in Figure 4 the sculptor with the efficient cause *f*, and the stone he is working on with the material cause *A* of the sculpture to be. The future sculpture is the product *B*.

Figure 4. An ancient sculptor and his work in progress

For convenience, we shall assume that the sculptor was illiterate and was not a painter. Therefore he had to work according to what he could picture in his mind if he did not have a life model, as well as the working procedures involved. Then it could be concluded that the sculptor as the efficient cause also physically represented the formal cause as indicated in Figure 4.

If the sculptor was literate and was a painter, the recording media of any type that held both the image of the future sculpture, and the working procedure if necessary, could still be considered as the representation of the formal cause. And the media in this case would be a physical extension of the sculptor. That is, the actual physical realization of the formal cause has no influence on the discussions of the entailment structure in any way. Figure 5 presented the three causes identified in Figure 4 in the graphical form of Rosen's abstract block diagram.

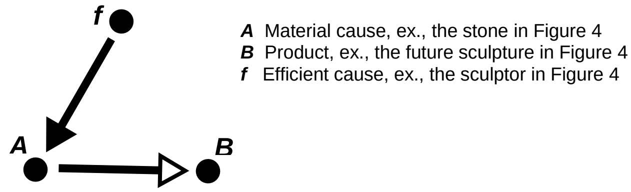


Figure 5. Entailment structure of the sculpture project in Figure 4

In Figure 4, the particular tools of the sculptor are also irrelevant with the discussions of entailment since they represent another physical extension of the efficient cause. Even after the hammer and chisel were replaced by modern equipment, the argument would still hold. Here again, the focus of efficient cause is to expose *the causal relation* between the product, i.e. the sculpture in this case, and who or what makes the material stone into the product sculpture.

Readers may now have discovered that Aristotle's causal framework offers us a unique opportunity for discussing relationships of entailment, which otherwise will be buried in material realizations [Rosen, 1985b and 1991]. No matter if it is a sculptor and his sculpture as a work-in-progress or anything else that is more complicated, any item listed in Table 1 for example, once it is possible to remove all specific material contents, they will share the same entailment structure as Rosen revealed in Figure 5.

Similarly, the reader will soon observe that even after the discussion on the finalities or final causes is involved, this material-independence inherited from Aristotle will still hold rigorously. The capability to study the pure relationships of entailment is an important reason why Rosen's system is fundamentally able to capture the big picture as completely as possible with scalability, as needed by enterprise development or systems engineering practice in general. In this series of articles, the readers will further find out that even an efficient cause can be purely relational. However, it is the author's experience in both academia and industry that the non-material nature of this relational thinking may well become an obstacle to understand this unconventional relational science for people with strong background of physical, digital, or material science: they used to think *How* to realized the relations instead of *What* these relations are about in the first place.

In order to maintain the rigor of the inferential process, it must be pointed out that once one introduces a high intelligence into the entailment structure, it becomes a necessary rule to associate the non-programmable intelligence directly with the efficient cause by definition. For example, the sculptor as an artist could never detail his creativity in advance as part of the formal cause. The creativity, the innovative power, and the aesthetic intuition can only stay with the sculptor who is the active change agent in this case.

The dual roles played by the sculptor as both the efficient cause and the formal cause represent another uncommon feature, entailment multiplicity, which is associated with the realization of Rosennean relational model. In the following discussions, the readers will find out that the sculptor and his sculpture in Figure 4 is only one of the examples to demonstrate the realization of the underlying entailment structure shown in Figure 5. Both the sculptor and the sculpture in progress are part of the model realization in this case. Even in this simple case, the mapping between the components of the entailment structure and the components of the model realization may not be one to one. Readers will find that every concept in this relational model may be relative since it is context-dependent.

It is time now to extend Rosen's theory into the field of enterprise development. As part of the preparation for further discussions, Table 1 presents the concepts of Aristotle's causalities based on Rosen's symbology, together with their mappings to some typical cases from the field of enterprise development.

Note that the International Task Force on Architectures for Enterprise Integration identified two types of Enterprise Reference Architectures [Williams, 1992; 1996; 1997; and 1999]. The Type 1 Architecture represents conventional architectures of equipment, computers or non-computers. That is why the first mapping case listed in Table 1 is one for production equipment, which is considered as a special type of enterprise accordingly.

Table 1. Aristotle's causalities presented in Rosen's theories and their sample cases from enterprise development

		Definition of Aristotle's Causalities[+]	Rosen's Symbol	Types of Enterprise			
				Operational Case		Developmental Case	
				Production Equipment in Operation	Enterprise in Operation	Housing Project	Enterprise Development Program
Product or Output Entity		(N/A)	*B*	Product	Product or service	House	Enterprise, business organization, or processes
Aristotle's Causalities	*Material Cause*	The substance, the resources to be consumed, or any environmental input, out of which the product entity, *B*, is produced	*A*	Raw materials, energy, etc.	Materials, energy, cash and information, etc.	Raw materials, Energy, etc.	Information, materials, energy, and cash, etc. i.e. all types of input as needed
	Efficient Cause	The change agent, or the effectors, that carry out the needed change to the input	*f*	Automatic equipment	Operations management, operators as needed, operational equipment	Builder, contractors as needed, construction equipment	Enterprise development teams of all disciplines as needed, equipment of all types as necessary
	Formal Cause	The structural features or attributes of a thing, or the plan and directions as needed, that the change agent follows	(Implicit*)	Built-in production process, software, operational and maintenance specifications.	Policies, and operational processes, control and management software, operational and maintenance specifications, etc.	Blueprints, construction and building standards, and engineering specifications, etc.	Program implementation methodology, program control and management software, program standards, engineering blueprints, and specifications, etc.
	Final Cause	The ultimate purpose, the source of the need, the end, or the goal of the product entity, *B*	Φ	Product user	End-user of the product or service delivered by the enterprise	Dweller of the house	Business management of the enterprise

[+]Note: The interested readers may find more discussions or references on Aristotle and his four causes at the online dictionaries of philosophy at http://www.philosophypages.com, or http://lgxserve.ciseca.uniba.it/lei/foldop/index.html. Particularly, Garth Kemerling, who contributes to both sites, presents an introduction and other associated links to the four causes at http://www.philosophypages.com/hy/2n.htm#causes, ©1997-2002 Garth Kemerling. [Rosen, 1985b] pointed out that instead of the cause-effect relation understood in commonsense, the Aristotelian causality should be understood as a way to interrogate the subject studied to explore the underlying reasons why the subject exists, or became a being. Other authors have also pointed out that there is no better English translation available for the four Aristotelian "causes", or "*aitia*" (singular, "*aition*") in Greek.

*Note: Rosen ignored an explicit presentation of the formal cause in his discussion, but readers may consider it is hidden behind the hollow arrow that shows the direction of the effect produced by the material cause, which is indicated in Figure 5 as ──▷. To the author's knowledge, this symbolic absence does not have any negative impact on Rosen's system.

In Table 1, the illustration starts with two operational cases of production or service entities, followed by two developmental cases of project entities, whose responsibilities are to create those operational entities. It shows that Aristotle's causalities are fully capable of describing the entailment structures of either operational or developmental project entities. The Task Force defined Type 2 Architecture as the description of the full life cycles of enterprise projects or programs, which are associated explicitly with the type of the project entities. In other words, as the reader will soon find out, Aristotle's causalities are capable of describing the generic entailment structures that underpin the enterprise reference architectures of the two different Types as defined by the Task Force.

The case of the housing project in Table 1 also plays two roles. One of them is to further demonstrate how the four causalities could be understood through common sense, based on what has been learned from the sculpture example. Another is to further help readers see the different perspectives between operational concerns and developmental concerns, a house and a housing project for example. The understanding of the former may not necessarily mean the understanding of the latter as Rosen once indicated [Rosen, 1991]. However, the appreciation of the difference between the two is a key to this extension of Rosen's system into enterprise development.

The pairs under each case of Table 1 help the readers visualize the increase of physical complexity when the discussions move from simpler entities to bigger systems. From the operations of a single piece of equipment to the operations of an enterprise as an organization, or from the housing project to the enterprise development program, the material manifestation of the four causalities will include more and more components. However, the increasing physical and associated conceptual complexity has no effect on the discussions if one stays with the Aristotle-Rosen relational entailment structure.

Given the cases of Table 1, let's start with the entailment structure of the production equipment in operation. By imitating Rosen's symbology in Table 1 and Figure 5, a simple model can be constructed as shown in Figure 6, where a one-to-one mapping between the terms involved with the equipment and those of Aristotelian causality is presented. That is to say, in the case of the equipment, the pairs of the terms such as *f* and Equipment, *A* and Raw Materials, and *B* and Product, are interchangeable.

Now with the abstract block diagram in Figure 6, for the production equipment of Table 1, one may explore its entailment structure by asking "Why *Product, B*?" A possible answer then can be "because *B* is made of the *Raw Materials, A*, i.e. the material cause," or "because *B* is produced by the *Equipment, f*, i.e. the efficient cause." One might even think by some tradition that the equipment here was at the same time the "final cause" of the *Product, B*. But that is self-referential in the logic structure of reasoning, while the true final cause Φ is sitting outside of the traditional discussion, which is the End-User of the *Product B* as shown on Table 1.

(a) Presentation in Rosen's symbols (b) Presentation in mechanical terms

Figure 6. Entailment structure of machine equipment presented under causal and mechanical terms

As the cases become more and more complicated, readers will see from Table 1 how uncomfortable it becomes if the efficient cause is kept as the final cause. Although some may still argue, the builder of the housing project could be the final cause of the house, not many people in the engineering field today will accept that operations management is the final cause of the enterprise in operation, or the development teams are the final cause of the enterprise in development.

The explanation for the causal relations may seem obvious by now. However if the readers are familiar with different types of definitions of enterprise integration, they will find out that PERA to date has been the only reference architecture that links enterprise integration directly and explicitly with enterprise management for business purposes [Li, 1995 and 2004; Williams, 1992], instead of with electronic connectivity [Bloom, 1997; Kosanke, 1999].

Rosen pointed out theoretically that the misleading logic associated with the entailment structure of machines comes from the missing entailment of the efficient cause *f*: what entails the efficient cause is graphically absent from Figure 6. In other words, the entailment structure of a machine does not tell anything about the object that entails the efficient cause *f*. This machine, as an artifact, does not have to know anything about the answer to "Why *Equipment*?" or "Why *f*?" in general, only because it is usually an external entity, such as a designer, or an end user, that generates a purpose for the artifact. For example, the patron of the sculpture in Figure 4 can be considered as such an entity. In terms of Aristotelian causalities, it is the final cause by definition as listed in Table 1 that answers this entailment question for the efficient cause.

The decision on externality or internality of the final cause for complex systems such as business enterprises may however generate unintended outcomes, which are usually overlooked by conventional analytical tools prepared for developing machines. The physical and digital structures of machines can be quite complicated, but under Aristotelian-Rosennean causal framework, these machines are simple systems as shown in Figure 6 in terms of their causal relationships as organization. The extrapolation of the entailment structures from the machines into enterprise life cycles will generate consequences that unfortunately surpass the reach of the machine model and are invisible under the tools that develop machines.

In order to illustrate the consequence of following the entailment structure of machines in developing enterprise life cycles, Rosen's theories and graphical tools will be applied first to conventional life cycles of enterprise development, which represent a detour to reveal the inherent limitation of machine model in enterprise development. Then, in comparison, the author will address the discovered inadequacy of the machine model and explore Rosennean entailment structures ingrained in the PERA life cycle.

4. Modeling Theories of Enterprise Development

In the case of the enterprise development program listed in Table 1, if the true final cause is excluded from the enterprise life cycle, a direct consequence will be that the life cycle must start with elicitation of requirements from the business management for the enterprise to be delivered. Then the efficient cause *f*, the development team(s) as listed on the last column of Table 1, will lead the life cycle and will act as the default "final cause" of the program instead of the business management. Since the machine model is known now, it can be understood why in the real world, many development teams consider themselves as the rightful representatives of the true final cause, the business management in question.

This common misstatement with many development life cycles does not go without notice by researchers in systems engineering. A regular remedy is for the development team involved to reach out actively and reinforce the connections with the true final cause outside the life cycles. For example, Sage made such sensible recommendations for a series of systems engineering life cycles as shown in Figure 7 [Sage, 1992], where RDT&E stands for Research, Development, Test And Evaluation.

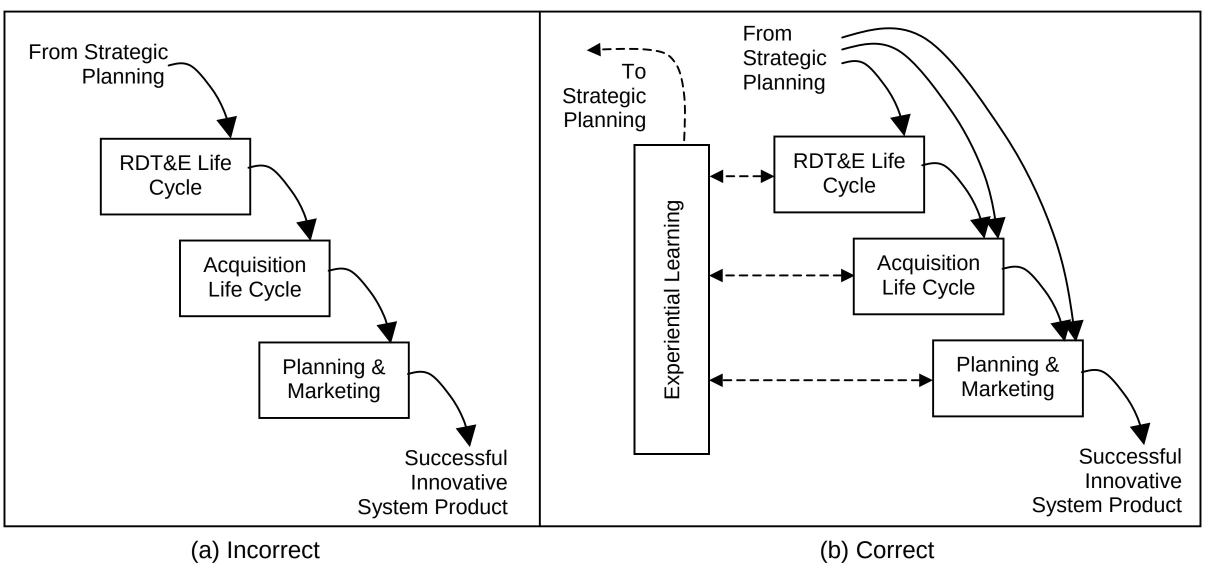
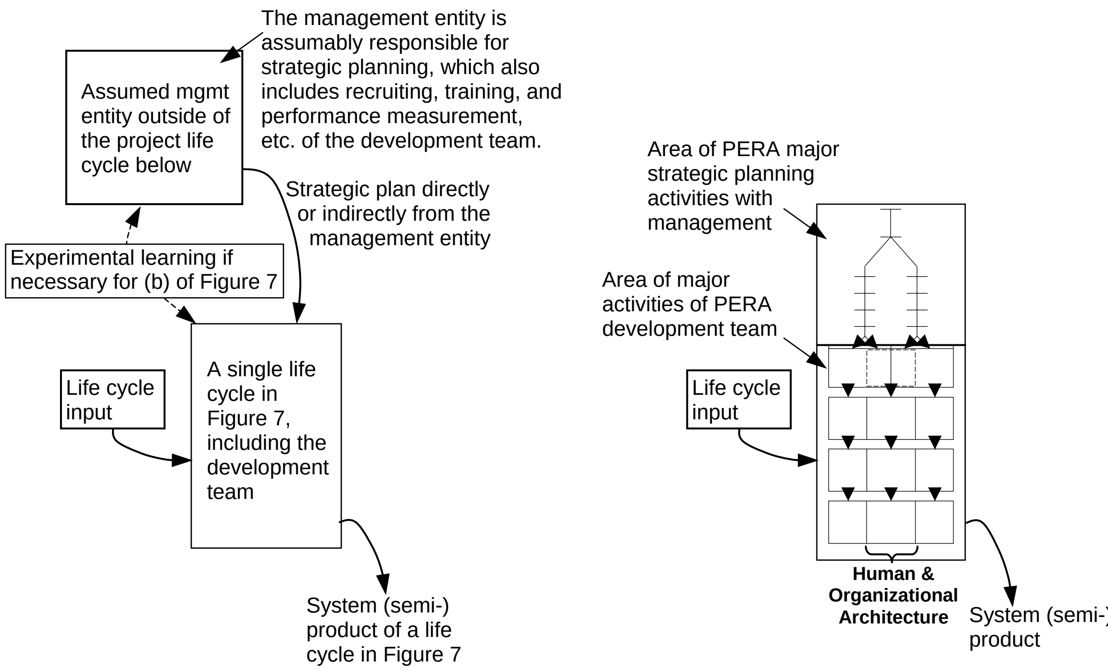


Figure 7. The "wrong" and "right" ways to picture life cycle interrelationships [Sage, 1992]

Each of the life cycles in Figure 7 receives a strategic plan from an outside entity. An assumption is that the strategic plan is originally created by an outside management entity where the activities of strategic planning in Figure 7 are conducted. In (a) of Figure 7, although the downstream life cycles such as Acquisition and Planning & Marketing may not receive directly the strategic plan as their counterparts in (b) of Figure 7 do, it is reasonable to assume that the upstream entities such as RDT&E and Acquisition will pass down the strategic plan together with the output of their life cycles. In addition, as indicated by Figure 7, each of the life cycles has its product, or semi product if it is not final yet for the end customers. By the time the life cycles start, they should have their own development teams ready.

(a) of Figure 8 presents such a typical life cycle model to prepare for further entailment study of the life cycles of Figure 7. In order to reveal the consequences of different organizational approaches to strategic planning, (b) of Figure 8 presents a simplified PERA life cycle model ready for entailment discussions later.



(a) A life cycle model without internal strategic planning capacity in Figure 7

(b) Simplified PERA life cycle model with internal strategic planning capacity

Figure 8. Exposition of causal relations in life cycle models

In order to further expose the entailment patterns in the life cycle models in Figure 7 and (a) of Figure 8, assume that the introduction of strategic planning into the life cycles indicates two types of connections between the development teams and the business management as follows, each of which has become common practice in industry:

1. Connection through team development that is part of the strategic planning process

   It is the business management that organizes the development teams according to the strategic plan as shown in Figure 7 and (a) of Figure 8. That is to say, the business management, $Management_{F7}$, is the efficient cause of the individual development team, $Team_{F7}$, involved. The candidates for the team members, $Candidate_{F7}$, are the material cause of the team. The strategic plan is the implicit formal cause. Note that the strategic planning process is in another organization where the business management $Management_{F7}$ is the efficient cause of the development team $Team_{F7}$, and both (a) and (b) of Figure 7 share the same team building process as shown in Figure 9.

   Another common pattern shared by (a) and (b) of Figure 7 is the relation between the development team and the product to be delivered, the enterprise as shown in Figure 10. Please note that throughout the project life cycles, as indicated in Table 1, the development team $Team_{F7}$ becomes the efficient cause of the enterprise to be, $Enterprise_{F7}$, as shown in Figure 10, where Input represents all physical and non-physical (intellectual, information, etc.) resources necessary for the development project in question as the material cause of the enterprise defined in Table 1. Environmental influence will also be included.

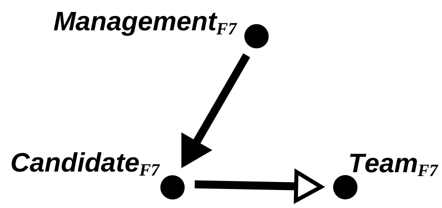


Figure 9. The entailment structure of team building of Figure 7

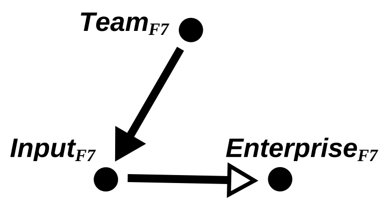


Figure 10. The entailment structure of team building of Figure 7

Since the entailment structures in Figures 9 and 10 represent all governing relationships of the project life cycles in (a) of Figure 7, the generic entailment structure of (a) can be shown in Figure 11 by simply adding Figure 9 and Figure 10 together. Therefore, the question "Why *Team*$_{F7(a)}$?" now can be answered with Figure 11 and subsequently the final cause of the deliverable, *Enterprise*$_{F7(a)}$, can be shown there as well. The answer clearly is that *Management*$_{F7(a)}$ is the final cause of *Enterprise*$_{F7(a)}$.

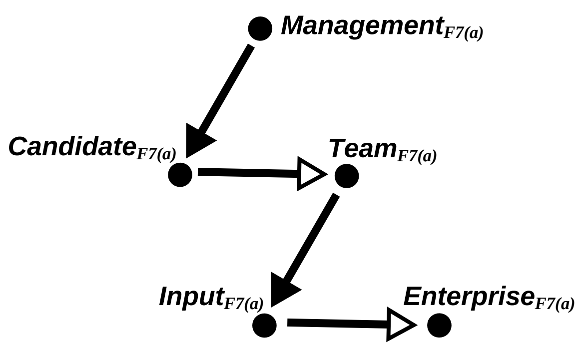


Figure 11. The relational life cycle model of Figure 7 (a) where entailment structure across two organizations presents the final cause of *Enterprise*$_{F7(a)}$

Please note that both the team and the strategic plan as a product package of the management entity in (a) of Figure 8 are passed down from the management entity to the development cycle where the team becomes the efficient cause of *Enterprise*$_{F7(a)}$ as shown in Figures 10 and 11. Since the strategic plan is implicit in the entailment pattern, the team can be considered a graphical representative of the strategic plan in Figure 10 and the lower part of Figure 11. Management in Figure 9 and the upper part of Figure 11 on the other hand may be also considered as a representative of the implicit strategic plan, however for a different role in a different entity: management is the creator of the strategic plan, whereas the development team is the implementer of the same plan in the implementation life cycle. The readers will find out soon that a clear separation between the creator and the implementer of strategies in such a top-down hierarchical manner may not always be the case, once various underlying causal relations are revealed. This clear-cut separation can be altered under either the machine model or the organic model, however with different consequences.

Although an answer to the final cause of *Enterprise*$_{F7(a)}$ is found, patterns in Figure 11 have created a new issue. The simple entailment structure, which is the same as the machine model in Figure 6, cannot answer the question "Why *Management*$_{F7(a)}$?" Sage pointed out correctly that the structure in Figure 7 (a) indicated an "incomplete" view of strategic management in life cycle management of a classical waterfall approach [Sage, 1992].

2. Connection back through the strategic planning process

As shown by the life cycle in (b) of Figure 7 and (a) of Figure 8, the development team may invoke communications as part of the learning process through strategic planning with business management, along with the development progress of the enterprise to be, $Enterprise_{F7(b)}$. Under the supposedly correct life cycles (Figure 7 (b)), the development team $Team_{F7(b)}$ in return becomes the efficient cause of the business management $Management_{F7(b)}$. This may sound strange that the development team "works on" the business management. But since the development team $Team_{F7(b)}$ is the action initiator that contributes informationally to $Management_{F7(b)}$ through the "Experiential Learning" box in Figure 7, they are an efficient cause of $Management_{F7(b)}$. $Enterprise_{F7(b)}$, or the work-in-progress if we could use the metaphor of the sculptor and his sculpture, becomes the material cause of the business management, as shown in Figure 12, since it is the source of the information. Again, the strategic plan as the formal cause is implicit here. This type of communication can be considered official in the strategic plan as indicated in Figure 7 (b).

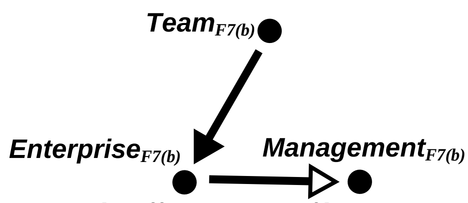


Figure 12. The development team as the efficient cause of business management shown by Figure 7 (b)

The entailment relation shown in Figure 12 is only part of the structure in the organization of strategic planning, because the elements of Figure 9 must also be included. The complete entailment structure of the strategic organization should be the one in Figure 13. The readers will immediately be able to tell that in this structure, *the business management and the development team are the efficient causes of each other*. At the same time, they can be the final causes of each other as well. By simply adding Figure 10 and Figure 13 together, Figure 14 represents the model of the complete entailment relations in Figure 7 (b).

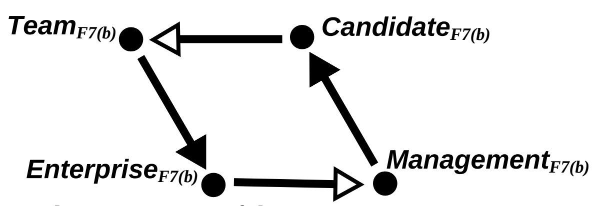


Figure 13. The entailment structure of the strategic organization in Figure 7 (b)

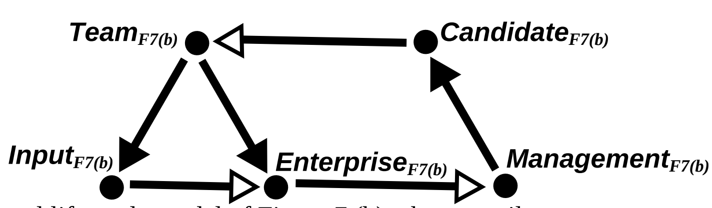


Figure 14. The relational life cycle model of Figure 7 (b) where entailment structure across two organizations presents the final cause of $Enterprise_{F7(b)}$

Now questions of "Why $Team_{F7(b)}$?" or "Why $Management_{F7(b)}$?" are answered. At the same time, the strictly hierarchical separation between the creator and the implementer of strategies in Figure 11 is replaced by reciprocal control between the two agents. However, the peculiarity of the self-referential structure in Figure 13 and Figure 14 presents at least three flaws in the entailment structure across the two organizations. These flaws can make the life cycle vulnerable to even small mistakes or environmental disturbance because the embedded entailment structure does not guarantee an effectively self-adjustable or self-learning capability. The congenital defects of the organizational structure are invisible to almost all available analytical tools today.

For example, the closed loop in Figure 13 does not represent a feedback control for the life cycle because of the lack of implementation. Instead, from the view of management communication, $Management_{F7(b)}$ trains and instructs the development team, $Team_{F7(b)}$, to present the information from their life cycle back to $Management_{F7(b)}$. If the preference from management was wrong and the development team followed the training exactly the way that management preferred, then management might never be able to detect anything wrong soon enough from the presentation prepared by the team, since they would mainly be checking that the team followed their preferences.

The feedback loop in Figure 7 (b) across the two organizations, *M-C-T-I-E-M* if we only keep the initials of the objects and ignore their subscripts in Figure 14, contains too many steps to be an efficient feedback control that can serve management. Any correction made by management must go through the team training and then be implemented by the team. The management outside the development organization may not know the results of the correction directly from the life cycle until the team pushes them back through the strategic planning process.

The double authorities assigned to the development team as efficient causes for both the enterprise and the management make the life cycle too vulnerable to potential mistakes made by the team, which is both the implementer of the life cycle and the controlling presenter of the results of the implementation. In other words, the team ultimately decides how the life cycle is conducted and how its results are presented. There is no way to stop the team from making honest mistakes in a situation where the team thinks it did the right job when it actually made mistakes.

By now readers may have learned something through the discussion above to see the unique strength of Rosen's relational approach with Aristotle's causalities. Unfortunately, the fundamental flaw rooted in the machine model that has been built into many enterprise systems and organizations today is still causing the same pain to the members of these systems/organizations. It does not matter where the members are in the hierarchies of the machine organizations, because they, who serve as the efficient causes, have no help from their organizational structures to understand the invisible root cause of their problems. Fundamentally and theoretically it is the machine model shown in Figure 6 that forbids them from knowing anything more than the machine allows, and it is the same machine model that has been hardwired into systems engineering conventions, which make the invisible organizational defects acceptable by common practice as indicated in Figure 7 (b).

Thanks to Robert Rosen, who not only shows us how to identify the problematic machine model, but also shows us how to fix it. Following Rosen's approach, by going through the detour in the typical world of machine models, we have found the problem by again asking the seemingly innocent question "Why *f*?" from the beginning. A simple question, but the machine-model cannot answer it. This leads the author to suggest that it is not the question "Why *f*?" that is absurd, but instead that the machine model, even equipped with the best calibrated feedback control, may still be too simple to be an effective support for the complexity management of enterprise development. Now our task is to address the inadequacy of the machine model, and create a new and more acceptable model for truly integrated human organizations.

That is to say, if we want our organizations to be able to adequately answer the question "Why *f*?" we will need to find this new model to make our organizations and our program life cycles inherently and organizationally creative, self-learning, and self-evolving, just like any biological organism is capable of doing through the internal abilities they have acquired during millions of years of evolution. Hopefully, we won't have to wait that long.

Rosen's abstract block diagram itself does not impose any unnecessary syntax. The entailment of the generic machine does not have anything to stop us from changing its structure either. For example, as indicated previously in Figure 3 and (b) of Figure 8, PERA master plan explicitly requires that the needed strategic planning be part of the life cycle. If we consider the enterprise to be as a work-in-progress during its life cycle as defined by PERA, we then should be able to pull the strategic planning process into the enterprise, instead of considering it part of another organization outside the development life cycle. In mathematical form based on Figure 14, we immediately should have *Candidate* as a subset of *Enterprise*, that is,

$$Candidate \subset Enterprise$$

And then after the subscripts are removed, Figure 14 should be modified as Figure 15. The entailment structure in Figure 15 indicates that the new life cycle of the enterprise, *Enterprise*, contains the strategic planning. Also business management, *Management*, must at least participate in the process of strategic planning within the organization, even though it is still officially an outsider of the organization. Although the structure in Figure 15 implies that the organization may still suffer the issues associated with problematic feedback processes as we discussed with Figure 13 and Figure 14, we know from our real world experience that the quality of any strategic planning process is always much improved if business management and the other members of the planning team become active participants.

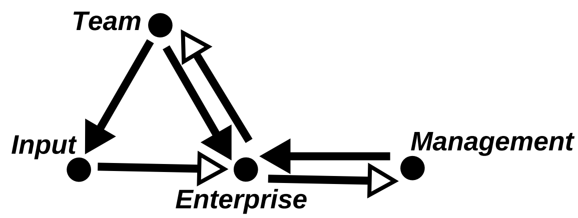


Figure 15. An enterprise whose life cycle includes its own strategic planning

However, the fate of business management hanging outside the development organization as well as the associated self-referencing loop in Figure 15 is still a defect in the entailment structure of the organization. Consequently it is necessary to break the flawed entailment structure where the development team acts as the efficient cause of business management.

Again, if we resort to the structure of PERA in Figure 3 and (b) of Figure 8, specifically the Human and Organizational Architecture, we can identify at least one approach to replace the defective relation, which is to establish a relation driven by business growth, instead of driven by the development team. In a sense, with this new relation, *Management* as well as the implicit strategic plan in Figure 15 can be considered as a "product" of the growth of *Enterprise*, and *Team* should act as the input, either information resource or candidate pool, in this special "production process". In other words, based on the structure shown in Figure 15, we now officially consider every object is within one organization and grows together with the enterprise. In business management terms, this practice is often referred to as "promote from within" if a member of the team is promoted to management.

In terms of the life cycle of the enterprise, at first glance, growth from within could mean that the members of the development team could be promoted into the management team of the enterprise to be. However, a more common practice will be to promote the members of operational teams of the enterprise at the Operations Layer shown in Figure 3. The Human and Organizational Architecture of PERA represents a device to make that possible either way. In this case, it is the business growth of the enterprise that drives this kind of managerial growth and strategic planning as shown in Figure 16.

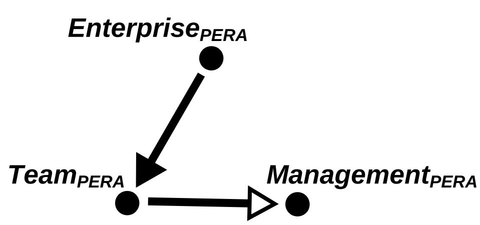


Figure 16. The entailment structure that is needed for growth of business management and strategic planning from within

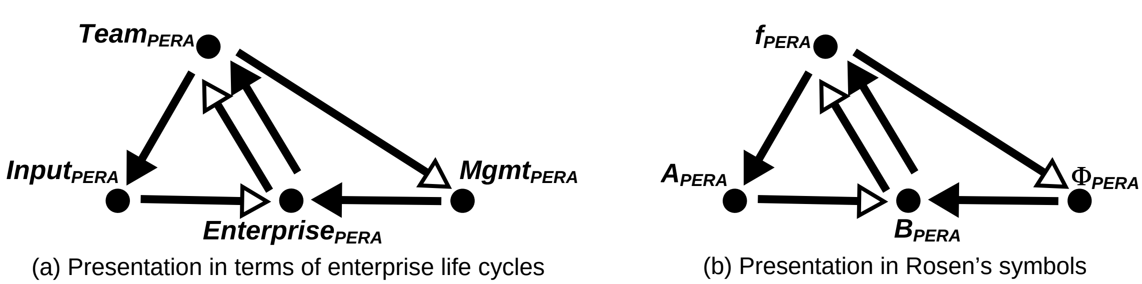


Figure 17. Relational model for the full life cycle of PERA enterprise

The change of the entailment structure made from Figure 12 to Figure 16 has turned the *Team* from the controller of enterprise information as shown in Figure 12 to the active contributor of the information in Figure 16, which has also blurred the clear separations between the creator and the implementer in the command-and-control structure in Figure 11. While the position of the *Management* as the information receiver is strengthened as a result, the growth of the *Enterprise* as a whole must also be committed to grow the *Management* by promoting the members of the *Team* into the enterprise management, and by nurturing an organizational culture that encourages the input from the *Team*. The corresponding entailment structure of the enterprise is then presented in Figure 17 (a). Since the organizational considerations of PERA are the conditions of the newly introduced entailment structures in Figure 16 and Figure 17, PERA is shown as subscripts there.

According to the mapping relations presented in Table 1, Figure 17 (a) can be expressed as Figure 17 (b) in terms of Aristotelian causalities. If readers compare Figure 17 (b) with Rosen's results as shown in Figure 18, it will not be difficult to find that the relational model of the entailment structure in Figure 17 is exactly the same as the one for organisms revealed by Rosen. In other words, we have proved through a series of experience-based "thought experiments" that PERA in theory can be a *necessary device* to grow an enterprise into a "Rosennean organism" using the entailment structure that the two share. QED.

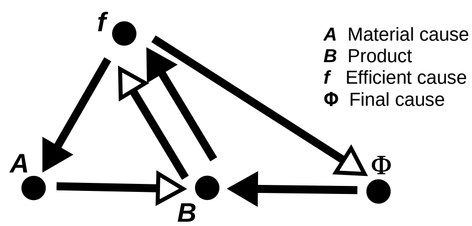


Figure 18. Rosen's relational model of organisms [Rosen, 1991, p. 251]

Please note in order to expose the hidden entailment structure, the previous discussions have simplified the relations around strategic planning between the Operational Team and the Enterprise Management. In a real world setting, when an enterprise becomes a cohesive whole as indicated by Figure 17, the contents of the relationship between the Operational Team and the Enterprise Management usually grow much richer than a formal strategic plan and its planning process could generate. The rich contents of the entailment relationships once established will enhance, not weaken, the arguments here in this article, because they demonstrate the potential of a truly organic institution [Selznick, 1957, pp. 5 – 6].

Although management or a strategic plan may officially declare the values of a corporate culture, the culture that promotes employee participation and career growth is usually much more than spoken or written acknowledgement in a document, and the establishment of value relationships between management and operations is not an overnight achievement. So called "manage by numbers" or "manage by data" therefore will have everything to do with the machine model, but nothing to do with this organic model if such value acknowledgement only stays on paper, instead of through managerial practice.

5. Discussions

Rosen points out that his relational model represents at least a necessary condition "*for a material system to be an organism.*" [Rosen, 2000, p. 28]. Nevertheless, Rosen shows us how to approach organisms through Aristotelian entailment relationships that he has revealed mathematically. That is, with help from Rosen and his theories, we have learned in the rich and colorful garden of enterprise development, one should never say, "a rose is a rose is a rose." Instead, it is important to find answers to the fundamental questions, "What is an Enterprise?" and "Why PERA?"

- *An enterprise development program represents human and organizational efforts to build a machine, or an organism, or something in between in order to fulfill certain purposes designated by responsible stakeholders.*
- *PERA theoretically is a synthetic device to help humans and organizations manage organized complexities throughout full life cycles of enterprises all the way through associated entailment relations, in the way Aristotle and Rosen predicted.*

In enterprise development, an organic enterprise is one that has built-in capability of strategic improvement and innovation, i.e. an inner capability of anticipatory adaptation to unforeseen influence for growth – as to all living things. When this inner capability is weakened, or ultimately removed, the adaptability of this enterprise will be reduced accordingly until it becomes a machine organization.

Please note that the above identification is based on "*the homology of system characteristics*" [von Bertalanffy, 1969, pp. 80 – 86] across seemingly different fields through *relational* abstraction, which is not a reductionist approach. Rosen described this type of abstraction strategy initially proposed by Rashevsky[4] as keeping the organization as a whole and throwing away all physical or material concerns [Rosen, 1985b; 1991; and 2000]. This abstracted organization turns out to represent Aristotelian entailment relations both in biology and in enterprise development study. The discussions of the latter are particularly based on the definition of the complete life cycle of PERA as shown in the previous sections.

From this viewpoint of relational modeling, Rosen points out the following, based on his studies in biology:

> *In this view, then, an organism is a material system that realizes a certain kind of relational structure, whatever the particular material basis of that realization may be. The trick, of course, is to find or posit that relational structure; this is not an empirical or experimental problem in any conventional sense.* [Rosen, 2000, p. 260]

If PERA did not insist on the completeness of the definition of enterprise life cycles in general, the hidden relational structure of Aristotle and Rosen inherent within enterprise development would probably never have manifested itself in this research. As shown in Figures 7, 11, and 14, the commonly acceptable practice in management and engineering has been to organizationally maintain two separate worlds, strategic and operational [Goranson, 2003; and Hammer, 2004]. The PERA life cycle does not necessarily make this artificial separation; it suggests instead an organic unification between the strategic and the operational by formally introducing the place of strategic development as well as human and organizational architecture into this generic enterprise architecture.

Table 2. Mapping between Aristotelian causalities and enterprise definition of PERA

Aristotelian Causalities	**Corresponding Items Defined by the Concepts Layer of Purdue Methodology**	**Note**
*Product (**B**)*	Program Business Plan Objective, Strategies and Goals Business Measurements of Program	• The high-level and comprehensive list provided by Purdue Methodology represents the major output of the Concepts Layer of PERA [Williams, 1996, p. 106]. • The Critical Success Factors presented in the Purdue Guide [Williams, 1996] can be part of any of the Aristotelian objects listed. • As required by PERA, the listed items represent business considerations. No specific means of implementation should be included here.
*Material Cause (**A**)*	Program Business Plan Business Measurements of Program	
*Efficient Cause (**f**)*	Program Ownership of Business Entity Program Business Plan	
Formal Cause	Program Business Plan Objective, Strategies and Goals Business Measurements of Program Purdue Methodology	
*Final Cause (**Φ**)*	Program Ownership of Business Entity Objective, Strategies and Goals Business Measurements of Program	

Rosen pointed out that, between the relational structure and the material realization, the latter will conceal the former once that realization is established [Rosen, 2000]. Please note that in enterprise engineering, as well as many other related engineering fields, the counterpart of "material realization" in the way used by Rosen is "physical implementation". As defined by the life cycle of PERA shown in Figure 3, the

[4] T. Gwinn at http://www.panmere.com compiled a list of books by Nicolas Rashevsky (1899-1972) including his work on relational biology: *Mathematical Biophysics: Physico-Mathematical Foundations of Biology*, Vol. 2, 3rd Ed. Dover Publications, 1960.

considerations of "how" to physically implement the enterprise entity in question start as early as the Specification Layer to specify the physical carriers (human or machine) of the defined enterprise functions, whereas the preparation for those functions starts as early as the Definition Layer.

Therefore, if the Definition Layer and the Specification Layer of PERA are considered part of the transformation process from the relational structure to the physical implementation during the life cycle of the subject enterprise, the first two Layers of PERA, Identification and Concepts, will play the major role in forming the relational model of Aristotelian causalities. The high-level and comprehensive Mission, Vision, Values, Strategies and Policies generated in the first two Layers of Purdue Methodology [Williams, 1996, pp. 103 – 106] should contain sufficient information to define the four causes of Aristotle and their causal relationships for the enterprise as an individual business entity. Table 2 shows such a mapping prepared based on the Purdue Guide [Williams, 1996] and the basic concept of Aristotelian causalities from Table 1.

Although Table 2 is not exhaustive, it illustrates the focus on the first two Layers of PERA, which are the questions and answers to "What to do?" and "Why?" without material concerns at the strategic level of the enterprise, just as predicted by Rosen when he discusses the relational structure of his M-R model with biological organisms.

Similarly, also predicted by Rosen, any premature discussions of physical realization or implementation will distract the business focus in these two Layers of PERA. No implementation concerns, including those of business processes and functions, should take place until the business identities of the subject enterprise, including business purposes, business value systems, business strategies, and major policies, have been properly clarified. Otherwise, in Rosen's word, these concerns of implementation may well "cloak" [Rosen, 2000] the true business considerations. However, the author's experience in research and industry has shown again and again that many professionals, particularly those with strong technical background, have difficulties grasping the relationally-oriented thinking when it is needed.

Guided by the Mission, Vision, and Values of Purdue Methodology, the nature of strategic planning is a process with an outward focus, not an inward focus. High-quality strategic planning is where the subject enterprise must first "*find itself, now and in the future*" within the business environment [Chambers, 1999, p. 14]. Therefore, the business views at the first two Layers, if desired by the business executives, should be those that directly relate to the market, competition, economy, legal, social and culture environments, etc. In Rosen's terms, it has to formulate an "*anticipatory system*" [Rosen, 1985b].

When the strategic world and the operational world remain separated, each of them will be imperfect models [Goranson, 2003]. Previous discussion in this article has demonstrated some imperfection of the separation between the two worlds by revealing the underlying machine models in both worlds. Conceptually it is "mechanical instructions" between the machine organizations, as exemplified in Figure 11 and Figure 14, that maintain the connection between the separated worlds.

The difference in the machine model and the organic model presents a big challenge to the common practice with modern management information systems. Without fundamentally changing the underlying model, any attempt to strengthen the connections between the "machines" may only receive limited payback through calibrations between the "machines", if the desired goal of the systems implementation is actually to establish flexibility and adaptability for the enterprise in a changing environment.

According to the relational model as shown in Figure 18, the machine model in Figure 6 only resembles the metabolic function of an organism. Although the metabolic function is essential, it is the repair and particularly the replicate functions of an organism (see Figure 19) that ultimately make it organic in order to grow, self-adapt, and self-evolve. However, theoretically, a common misunderstanding of the three organic functions defined in the M-R model is to treat all three as the simple equals under the abstract concept of morphism without proper discrimination, as if they were just variations of metabolism that could be manipulated by the laws of physics and chemistry. For example, [Goertzel, 2002] and [Landauer, 2002] both accepted Rosen's definition of metabolism in the M-R model. They however manipulated and consequently changed, to different degrees, the mathematical definition of replicate. Without proper descriptive languages to connect their formulas to organisms, their new formulas have become merely a mathematical exercise, not representative of what the repair and replicate functions really accomplish.

Organically, it is the metabolic function as shown in Figure 19 that is responsible for transforming the physical and chemical input into the output that maintains the basic needs of life, which is comparable to machine function. Associated process management and control of machines should be included within the metabolic function as well. The repair function in Figure 19 is responsible for regenerating and repairing the entailment structure of the metabolic function, and the replicate function is in turn responsible for regenerating the entailment structure of the repair function. Therefore, their realization does not exist in mere material form, like metabolism. They manifest largely as relations in Rashevsky's relational term. [Klir, 2001] called the former "thinghood", the latter "systemhood."

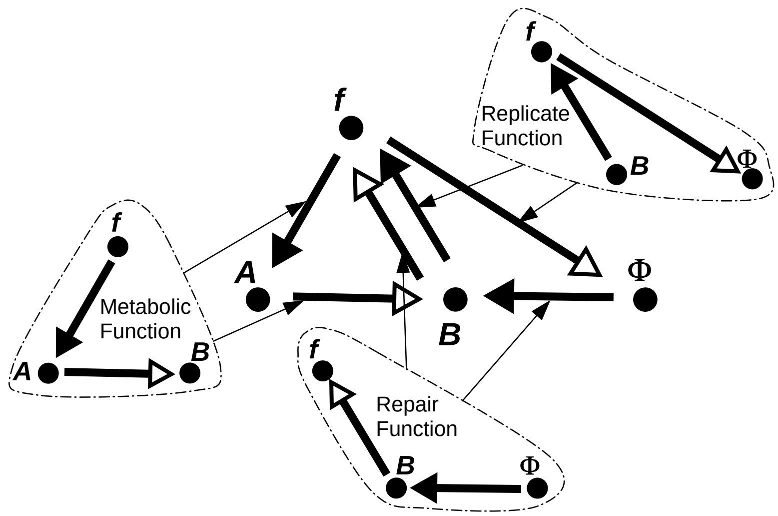


Figure 19. Three categories of morphisms within the M-R model

Both the repair and replicate bio-functions have had no full material realization in industry or engineering except in some computerized simulative computation. While Rosen pointed out many times that his M-R model is inherently non-computable, he did not reject the possibility of simulation. He rejected a full-fledged materialization of the M-R model under reductionism. In Gödel's language, the overall semantics of the realization of the M-R model are much more complicated than the semantics that realize metabolism. Major life activities may be studied after they are reduced into physical or chemical activities, but only in the context of metabolism. Although similar studies can be carried out for the repair and replicate functions, Rosen stated again and again that the current semantics offered by modern physics are still too impoverished to decode the secrets of life.

Please note that three separate presentations of the three bio-functions in Figure 19 do not imply that the implementation of the M-R model could break the model up into three functional pieces and then realize them piece by piece in a reductionist manner. To be an organic organization, the three functions must form and grow as a coherent whole. Unfortunately, the siloed practice in enterprise development currently still pays little attention to the necessary links between the three functions. For example, it is common that the links between operational data, engineering data, and strategic data of an enterprise are kept minimal. In order to mend the missing links, the Purdue Master Plan, as shown in Figure 3, places focus on the repair and replicate functions, as shown in Figure 20.

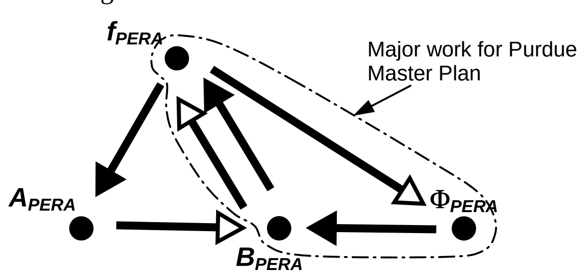


Figure 20. Focal area of Purdue Methodology

Please note that even the comprehensive PERA can be broken up according to the machine model, and then implemented in a reductionist mode, which is exactly where the attempt to automate PERA has led. With help of the relational framework from Rosen, it is not difficult to understand that the implementation of the machine model may be justifiable under stable and familiar business conditions, since it promises high efficiency as long as everything proceeds according to the original plan. That is to say, under this special situation, the Rosennean organic model may be simplified into a machine model, or simply reduced into metabolism. Due to the importance of metabolic function demonstrated by the M-R model in Figure 19, we may say that *theoretically there is a living machine in every living organism*. However, the same model is also telling us that it is not proper to reduce the organism into this machine when we need a living being, because it will immediately kill the life of the organism.

It is interesting to see that the most important strategic values of the machine model are fulfilled when it represents the metabolic function that has become such a living machine. In the context of enterprise development, it is where the operations team becomes an active and necessary contributor to the strategic growth of the organic enterprise.

In the field of enterprise development, the inclusiveness of the holistic organic model actually makes the reductionist approach more than an icon of simplification. For example, the iterative and incremental growth of the organic development will inevitably need reductionist contribution to develop small enough building blocks made from machines, instead of large monolithic ones that will make organic options of incremental implementation impossible. The applications of the organic model in return will provide anticipatory and experimental conditions that may offer minimum risk and best effectiveness available to experiment with those small blocks to explore the changing environment.

6. Summary

Rosen's definition of life and his relational model, M-R model, should remind the current research and practice in many fields outside biology to rethink the basic concepts of organic organizations and organic growth. In modern enterprise development, merely iterative and incremental development without close business orientation, or without embracing the importance of internal human and organizational development, are still not very far from the machine model that offers limited adaptability and flexibility. Unfortunately, the actual paradigmatic heritage may well be invisible if people involved do not recognize the underlying entailment structure of their organizational approaches. It is easy to overlook the repair and replication, which means the integration between the three organic functions, metabolism, repair, and replicate, is also lost. While many enterprises today may claim that they embrace an organic approach for their projects, what they have actually implemented is still a simplified metabolic equivalent, the machine model as shown in Figure 21.

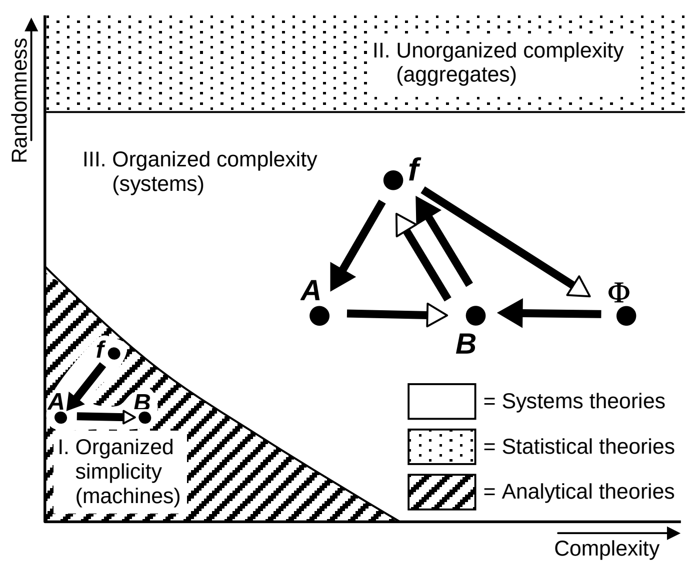


Figure 21. Machine and organic models for complexity management

Although reductionist thinking has helped machine development for centuries, it may well become one of the most difficult mental barriers to understanding Rosennean organic development, unless its limits are fully recognized and its relations with the holistic organic model are acknowledged. Only by then, the inclusiveness of Rosennean relational model can be fully appreciated as indicated in Figure 21 as an effective tool to deal with the organized complexity.

Enterprise information is a crucial part of any management decision process, and therefore should not be regarded as being operational without any strategic value. However, a single-minded reductionist approach will offer us an information system that is nothing more than an equivalent of mechanical equipment [Carr, 2003]. Under the machine model without internal strategic orientation, even the most advanced information systems may only enhance the connections between machines. These machine enterprises will stay put until they are forcefully changed by external intervention or at best by externally preplanned adaptation.

The key to fully releasing the potential values of information lies deeply inside the relations or entailment structures between business initiative and physical implementation, as this PERA research as well as many business professionals in the field have already discovered. However, since the needed theoretical support is largely missing in action, those successful business professionals (Tom Peters, Peter Drucker, Jim Collins, Jack Welch, Ken Blanchard, Peter Senge, Michael Gerber and many others) and their projects have to rely heavily on individual creativity, just as artists do. This reliance on individual creativity and heroics is risky in industry. It's hit-or-miss. It is our scholarly duty to help practitioners transform these risky artistic approaches into predictable craftsmanship by supplying the missing theories, such that the successful practice will be repeatable and scalable. This research has found that Rosen as a theoretical biologist has offered us his helping hand to manage organized complexity in enterprise development, represented by his M-R model in Figure 21. An organic infrastructure for enterprise development in Rosen's sense is realistic.

In addition to the metabolic function, the generic formal cause within the Aristotle-Rosen framework has illustrated in theory where enterprise modeling should focus. This points out a largely unstudied area for enterprise development, where the mathematical theories involved are familiar to many modern computer scientists. Even the reductionist approach will become more meaningful once its relationships with holistic approach revealed by Rosennean organic model are restored. The strategic aspects of this study for organic information systems do not need the author to prove their values at all, because the synonym of *entailment* is *business alignment*.

Acknowledgments

The author very much appreciates the valuable comments and suggestions of Theodore Williams with Purdue University, Daryl Kulak with Simplicity Institute, Shelby Laurents with Fluor Corporation, and Judith Rosen with Rosen Enterprises through numerous revisions of this article. The author is very grateful for insightful help from Judith to understand her father, Robert Rosen's life-long work.

7. Appendix A Philosophical Understanding of Reductionist Approach

A reductionist approach represents an attempt or tendency to explain complex fact, entity, phenomenon, or structure solely by its smaller pieces: it is expected that the whole can be re-assembled with nothing else but these smaller pieces. (see *The American Heritage Dictionary of the English Language*, 4th Edition, Houghton Mifflin Company, 2000). Interested readers may reference the discussions of methodological holism vs. reductionism from Healey, Richard, "Holism and Nonseparability in Physics", *The Stanford Encyclopedia of Philosophy*, Fall 1999 Edition, Zalta, E. N., Ed. available at http://plato.stanford.edu/archives/fall1999/entries/physics-holism.

The author has found that the conventional explanations of reductionism or reductionist approach in common philosophical discussions are usually contradicted against the concept of holism or holistic approach as typically shown in Healey's discussions referenced above. Please note however that this research demonstrates that a machine model as a reductionist model is a subset of Rosennean organic model, which is a holistic model. In other words, the two approaches are not necessarily exclusive from each other once the inclusiveness of the holistic approach is identified under Rosennean organic model.

The idea of Multimethodology that recognizes contributions from different philosophical paradigms has been studied and populated for years in management science, operations research, and systems science [Jackson 2000; Mingers, 1997 and 2001]. The reductionist approach can be classified under the *hard* methodologies of this "*methodological pluralism*" [Mingers, 1997, p. 2]. [Brocklesby, 1997] points out that the transitions between different paradigms can be difficult. He suggests that the multimethodological approach itself may need a systems paradigm to help the learning process. It is the author's opinion that Rosennean organic model with its rich perspectives may become a candidate for this meta-systems paradigm.

8. Appendix B Basic Syntax of Rosen's Abstract Block Diagram (ABD)

Rosen adopted the name, Abstract Block Diagram, at least for two different graphic tools during his career. The following presentation is summarized by the author based on the one that he prepared for his M-R model in [Rosen, 1991].

8.1 Two Basic symbols of ABD

i. An object represents an Aristotelian cause or effect, i.e., material cause, efficient cause, final cause, or product. An object is presented by a solid dot in ABD.
ii. An arrow represents the causal direction from a cause to the effect. There are only two types of arrows, one with solid head and one with hollow head.

8.2 Three basic rules of ABD

i. An object in ABD may play multiple roles of Aristotelian cause/effect.
ii. An arrow with solid head always starts from an efficient cause to a material cause. An arrow with hollow head always starts from a material cause to a product.
iii. Aristotelian formal cause is implicit in ABD.

8.3 Two corollaries of ABD

i. Whenever an arrow with solid head comes in, the object must always direct an arrow with hollow head to another object in the diagram.
ii. The basic building block of ABD is a combination of three objects connected by two arrows, one with solid head and another with hollow head such as

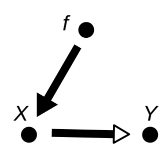


Figure A1. Basic building block of ABD

8.4 Mathematical expression of ABD

In categorical language, the basic building block of ABD in Figure A1 can be presented as

$$f : X \to Y$$

where
X – Domain, or material cause in Aristotelian causal language
Y – Codomain, or product in Aristotelian causal language
f – a map, or efficient cause in Aristotelian causal language

9. References

Bernus, P, Nemes, L., and Williams, T. J., Eds. (1996) Architectures for Enterprise Integration, Chapman and Hall, 1996.

Bloom, H. M. (1997) "Enterprise Integration – A United States View," Enterprise Engineering and Integration: building international consensus, Proceedings of ICEIMT '97, International Conference on Enterprise Integration and Modeling Technology, Torino, Italy, October 28-30, 1997, Kosanke, K. and Nell, J. G. Eds., Springer, 1997, pp. 6 – 19.

Brocklesby, J. (1997) "Becoming Multimethodology Literate: an Assessment of the Cognitive Difficulties of Working Across Paradigms", in [Mingers, 1997], pp. 189 – 216.

Brown, J. R. (2000) "Thought Experiments", The Stanford Encyclopedia of Philosophy (Summer 2002 Edition), Zalta, E. N., Ed., available at http://plato.stanford.edu/archives/sum2002/entries/thought-experiment/.

Carr, N. G. (2003) "IT doesn't matter", Harvard Business Review, Vol. 81, No. 5, pp. 41 – 49.

Chalmeta, R., Campos, C., and Grangel, R. (2001) "Reference Architectures for Enterprise Integration", Journal of Systems and Software, Vol. 57, No. 3, pp. 175 – 191.

Chambers, L. D. and Taylor, M. A. P. (1999) Strategic Planning: Processes, Tools, and Outcomes, Ashgate.

Goertzel, B. (2002) Creating Internet Intelligence, Kluwer Academic/Plenum Publishers, New York.

Goranson, H. T. (2003) "Architectural support for the advanced virtual enterprise", Computers in Industry, Vol. 51, No. 2, pp. 113 – 125.

Goranson, H. T. (1992) "The Suppliers' Working Group, Enterprise Integration Reference Taxonomy," Enterprise Integration Modeling, Proceedings of the First International Conference, Petrie, C. J., Ed., The MIT Press, Cambridge, Massachusetts, London, England, 1992, pp. 114 – 130.

Hammer, M. (2004) "Deep Change, How Operational Innovation Can Transform Your Company", Harvard Business Review, Vol. 82, No. 4, pp. 84 – 93.

ISO 15704 (2000) Industrial automation systems - Requirements for enterprise-reference architectures and methodologies, International Standards Organization, Geneva, 2000, a draft version is available at http://www.mel.nist.gov/sc5wg1/gera-std/15704fds.htm.

Jackson, M. C. (2000) Systems Approaches to Management, Kluwer Academic/Plenum Publishers, New York.

Klir, G. J. (2001) Facets of Systems Science, 2nd Edition, Kluwer Academic/Plenum Publishers, New York.

Kosanke, K., Vernadat, F., and Zelm, M. (1999) "CIMOSA: Enterprise Engineering and Integration," Computers in Industry, Vol. 40, No. 2-3, pp. 83 – 97.

Kosanke, K. (1996) "Comparison of enterprise modeling methodologies", Information Infrastructure Systems for Manufacturing, Proceedings of the IFIP TC5/WG5.3/WG5.7 international conference on the Design of Information Infrastructure Systems for Manufacturing, DIISM '96, Goossenaerts, J., Kimura, F., and Wortmann, H., Eds., Chapman & Hall, London, Weinheim, New York, Tokyo, Melbourne, Madras 1997. A bridged version of this article is available at http://cimosa.cnt.pl/Docs/cmm.htm.

Landauer, C. and Bellman, K. (2002) "Theoretical Biology: Organisms and Mechanisms", AIP Conference Proceedings Vol. 627(1), September 2, 2002, pp. 59-70.

Li, H. and Williams, T. J. (2004) "A Vision of Enterprise Integration Considerations", 2004 International Conference on Enterprise Integration and Modelling Technologies (ICEIMT '04), University of Toronto, Canada, October 9-11, 2004.

Li, H. and Williams, T. J. (2000) "The Interconnected Chain of Enterprises as Presented by the Purdue Enterprise Reference Architecture", Computers in Industry, Vol. 42, Nos. 2-3, pp. 265–274.

Li, H. (1995) A Formalization and Extension of the Purdue Enterprise Reference Architecture and the Purdue Methodology, Ph.D. Thesis, Purdue University, West Lafayette, Indiana, December 1994. Also published as Li, H. and Williams, T. J., A Formalization and Extension of the Purdue Enterprise Reference Architecture and the Purdue Methodology, Technical Report 158, available at http://iies.www.ecn.purdue.edu/IIES/PLAIC/PERA/Publications, Purdue Laboratory for Applied Industrial Control, Purdue University, West Lafayette, IN 47907, April 1995.

Mingers, J. (2001) "Multimethodology - Mixing and Matching Methods", in Rational Analysis for a Problematic World Revisited, Uncertainty and Conflict, 2nd Edition, Rosenhead, J. and Mingers, J. Eds. John Wiley & Sons, Chichester, pp. 289 – 310.

Mingers, J. and Gill, A. Eds. (1997) Multimethodology, The Theory and Practice of Combining Management Science Methodologies, John Wiley & Sons, Chichester.

Ortiz, A., Lario, F., and Ros, L. (1999) "Enterprise Integration – Business Processes Integrated Management: a proposal for a methodology to develop Enterprise Integration Programs", Computers in Industry, Vol. 40, Nos. 2 – 3, pp. 155 – 171.

Patankar, A. K., and Adiga, S. (1995) "Enterprise Integration Modeling: A Review of History and Practice", Computer Integrated Manufacturing Systems, Vol. 8, No. 1, pp. 21 – 36.

Rolstadås, A. and Andersen, B., Eds. (2000) Enterprise Modeling, Improving global industrial competitiveness, Kluwer Academic Publishers, Boston, Dordrecht, London.

Rosen, R. (2000) Essays on Life Itself, Columbia University Press, New York.

Rosen, R. (1991) Life Itself, Columbia University Press, New York.

Rosen, R., ed. (1985a) Theoretical Biology and Complexity, Three Essays on the Natural Philosophy of Complexity Systems, Academic Press.

Rosen, R. (1985b) Anticipatory Systems, Pergamon Press, New York.

Rosen, R. (1978) Fundamentals of Measurement and Representation of Natural Systems, Elsevier.

Rosen, R. (1958) "A Relational Theory of Biological Systems," Bulletin of Mathematical Biophysics, Vol. 20, pp. 245 – 260.

Sage, A. P. (1992) Systems Engineering, John Wiley and Sons, Inc., New York.

Selznick, P. (1957) Leadership in Administration, A Sociological Interpretation, Row, Peterson and Company, Evanston, Illinois.

Simon, H. A. (1977) The New Science of Management Decision, Revised Edition, Prentice-Hall, Inc., Englewood Cliffs, N.J.

Suh N. P. (1999) Axiomatic Design: Advances and Applications, Oxford University Press, New York.

Suh N. P. (1990) The Principles of Design, Oxford University Press, New York.

Uppington, G. J. (1998) Identifying the Road Ahead: Enhancing and extending the Enterprise Integration Identification Phase, Ph.D. Thesis, Griffith University, Brisbane, Australia.

Vernadat, F. B. (1996) Enterprise Modeling and Integration: principles and applications, Chapman and Hall, London, Winheim, New York, Tokyo, Melbourne, Madras.

Von Bertalanffy, L. (1969) General System Theory, Foundations, Development, Applications, Revised Edition, George Braziller, New York.

Weaver, W. (1948) "Science and Complexity", American Scientist, Vol. 36, pp, 536 – 544; selected by [Klir, 2001], pp. 533 – 540.

Weinberg, G. M. (1975) An Introduction to General Systems Thinking, John Wiley and Sons, New York.

Williams, T. J. and Li, H. (1999) "PERA and GERAM–Enterprise Reference Architectures in Enterprise Integration," Information Infrastructure Systems for Manufacturing, Mills, J. J., and Kimura, F., Eds, Kluwer Academic Publishers, Norwell, MA, pp. 3 – 30.

Williams, T. J. (1998) "Characterization of the Place of the Human in Enterprise Integration" In Handbook of Life Cycle Engineering, Concepts, models, and technologies, Molina, A., Kusiaka A., and Sanchez, J. Eds. Kluwer Academic Publishers, Dordrecht, Boston, London, 1998.

Williams, T. J. and Li, H. (1997) "The Task Force Specification for GERAM and its Fulfillment by PERA," Annual Reviews in Control, Vol. 21, pp. 137 – 147.

Williams, T. J., Rathwell, G. A., and Li, H., Eds. (1996) A Handbook on Master Planning Implementation for Enterprise Integration Programs. Based on the Purdue Enterprise Reference Architecture and the Purdue Methodology, available at http://iies.www.ecn.purdue.edu/IIES/PLAIC/PERA/Publications, Purdue University, West Lafayette, Indiana, US.

Williams, T. J. (1992) The Purdue Enterprise Reference Architecture, Instrument Society of America.

Williams, T. J. (Ed.) (1989) Reference Model for Computer Integrated Manufacturing, Instrument Society of America, Research Triangle Park, NC.