Paper #kdp018

An Application to Systems Engineering of a Framework of General Schemas Theory

Kent D. Palmer, Ph.D.
PO Box 1632, Orange, CA 92856
kent@palmer.name

Abstract

This paper is concerned with the application of a new framework for Systems Engineering developed under the rubric of General Schemas Theory. General Schemas theory is an extension of Systems Theory. Systems Theory is the logical academic foundation of Systems Engineering Practice. However, in the process of exploring the usability of this foundation it was discovered that Systems Theory needs to be extended into a General Schemas Theory so that it may prove be more useful as a basis for Systems Engineering, and in fact, Systems Engineering needs to be thought of as a *Schemas Engineering* in order to fit into this new context.

A 'System' is just one schema among many that Systems Engineers can use to understand the problems they face as they design solutions to those problems. There are other schemas such as Form, Pattern, Meta-system[1], Domain and World. All of these schemas, as well as some others, form a hierarchy of templates of understanding which can be useful to Systems Engineers as they design and build ever more complex configurations of elements which, perhaps, are not adequately described by the 'system' schema alone. In fact, Systems Theorist George Klir[2], among others, has taken to producing Advanced Systems Theories that combine several schemas into a single approach. But, as yet, no one has surveyed the entire field of all schemas and suggested a discipline analogous to General Systems Theory that would study the relations between all the different schemas that are possible. This paper has evolved from a research project which has exactly that goal. It suggests a set of canonical schemas found throughout the scientific disciplines in various incarnations, it suggests a way of thinking about the relations between these schemas, and most importantly it considers a framework that encompasses the hierarchy of schemas and augments it in order to establish an advanced conceptual framework in which we might reframe Systems Engineering practice. But this advanced conceptual framework needs some explanation because it suggests new ways of conceptualizing systems and associated configurations of elements. In this paper we will present schemas within this framework and attempt to discuss the practice of how the framework might be applied to Systems Engineering in order to improve the state of the art. Unfortunately this new framework is fairly sophisticated and takes us into territory that is unfamiliar to most classically (i.e. industrially) trained Systems Engineers.

[1] I now call the meta-system schema an "open-scape". But the usage will not be corrected in this paper.

[2] http://bioeng.binghamton.edu/faculty/klir.html

Considering the philosophical and scientific basis of our methods is not a normal activity within the Systems Engineering discipline. This discipline arose from *industry* and has only recently begun to put on academic airs. Many working systems engineers are suspicious of this academizing of their practical discipline. But, on the other hand, some systems engineers are worried that the foundations of their discipline are not clearly established. When we look into those foundations we discover that in order to clarify them we need to remake the framework within which Systems Engineering seeks its own foundations. Strangely enough, we need to stop talking only about systems, because we have come to realize that if we call everything a 'system' then the term becomes meaningless. We must distinguish between the different categories of schemas if, for no other reason, than to give the term 'system' its own meaning in relation to other possible schemas. But when we distinguish the other schemas we realize that we need those too and that we cannot do with just the system schema after all! Rather, as our systems become more complex, they break out of the bounds of the system schema and introduce us to the vagaries of all different manner of schemas which interact in complex ways. It is this complex interaction of the schemas that is the target of the problem solving and design activities that we wish to address with General Schemas Theory.

Introducing The Hierarchy Of Schemas

In his book Architecture of Systems Problem Solving[3] George Klir combines three different schemas to produce an advanced General Systems Theory. But, if we combine various schemas together, by rights we should call this General Schemas Theory based on the assumption that all Schemas should be treated equally in our consideration. The point is that there is a whole hierarchy of schemas that goes beyond the system, form, and pattern used by George Klir in his advanced theory. Instead we suggest the following ontological emergent hierarchy of schemas:

- Pluriverse
- Kosmos
- World
- Domain
- Meta-system[4]
- System
- Form
- Pattern
- Monad
- Facet

This series of schemas establishes the relation of phenomena to the human scale[5]. They are templates of understanding for phenomena that presents itself. They are a first categorization of all phenomena since everything that appears must appear in one of these schemas. They are templates of understanding because they are the first unconscious attempt at preparing the phenomena to be understood by schematizing it into one of these kinds of organization into spacetime envelopes. Once a phenomenon has been assigned a template of understanding it is possible to begin to come to terms with it by discovering its essence, the categories it belongs to, its unique characteristics, and its significance. If we reverse this process as we do when we design something new, then the schemas become the anchor of everything we design. It is the means by which we specify an embodiment within an envelope of spacetime for each part of the design. Individuation and

[3] (1985; New York : Plenum Press)

[4] *Open-scape* which is the combination of Meta-system and Infra-system. See "Towards a Possible Approach to Metasystems as Escapements" at http://holonomic.net

[5] This goes back to the declaration of Protagoras that man is the measure of all things. It is through the schemas that the first approximation of this measure is made.

categorization of design elements ultimately serve the schema, because if there is no schematization there can be no embodiment. So, as Systems Designers, who see our work as a process of implementation, schemas are very important to us. These schemas are hidden from us in normal practice, because we already know that what the schemas embody is implicitly understood as tacit knowledge. This is so much the case that we never focus on schemas as such. Our schematization process remains unconscious to the extent that we do not name the schemas that we use every day. We only talk about *kinds of things* as if essence was the only constraint on things. However, embodiment exerts other constraints on things than merely *kindness*; the schema that something inhabits is one of those very basic characteristics of all things which is least remarked on, but which is completely different from its essence. In philosophy we talk about the difference between essence and existence. Schematization lies between these two. Like the essence of the thing, a schema is part of Being. It is a part of its *Thisness* or *Thatness* as a spacetime envelope which allows the thing to be referred to. *Existence* per se is different from *reference*. Schemas allow the possibility of making a reference to something that exists in a spacetime envelope. Existence has to do with whether the thing is found or not. It is different from the aspects of Being which are *identity*, *reality*, *presence* and *truth*. These will become important later in this essay. At this point we will merely define existence as that which is neither *aspect* nor *anti-aspect*. That which is *both* aspect and anti-aspect we will call *quintessence*. Essences are combinations of the various aspects and anti-aspects of Being. The quintessence is the anamorphic and paradoxical combination of all aspects and anti-aspects at once. Essences define kinds of things and are constraints on instantiated attributes of particulars. The Essence applies to what inhabits a spacetime envelope. But the establishing of the spacetime envelope is independent of the identification of the essence, or even the individual unique characteristics of the thing. It goes beyond the definition of the essence and further beyond the signification of a thing by way of projected interpretation. Schemas are the building blocks of the embodiment of everything but we hardly notice them because we are so enamoured with the essences, the unique characteristics, and the significations of things.

A Representation Of The Schemas Hierarchy

We can relate the schemas (which are templates of understanding for things) to a particular mathematical object in order to specify its definition further. When we do that we are producing a representation of the schema. In this case our mathematical object will be the *Triangle of Pascal*. This triangle is built by adding the numerical results of one row together as a pair to produce the next row. So the triangle is produced by the repetition of addition, and it is always an addition of all pairs in the sequence of the last row which produces or begets the next row. The series of numbers generated is always a palindrome and it has the value of 2^n when all the numbers are added together in the same row. This sequence generates the minimal solid with n points of n-1 dimension embedded in each dimension. It also records the possible permutations of polynomials.

The key point here is that the Triangle of Pascal is a kind of dual of the schemas where each schema occupies two dimensions, with two schemas per dimension. This unusual fact suddenly makes the idea of the schema very concrete because we can test our hierarchy of schemas against this mathematical structure to see whether or not it has this form or some other form. Schemas are not reducible to dimensions, they are templates of understanding, and they are governed by a

dimensional hierarchy in respect to their relations with each other and to the things which are embodied by the various schemas. Thus the question that can be asked is: Why is there a double duality concerning the relation of schemas to dimensions?[6] We speculate that it is because there is a communication of representations at each level of the ontological hierarchy as well as a dimensional transformation of each schema across dimensions. Going *down* the series of schemas toward dimension zero results a in Representational information loss. Going *up* requires what Deleuze calls Repetition, which is the opposite of representation. Through repetition each schema arises as a *sui generis* emergent event. The repetition of information at the lower level never quite adds up to the emergent characteristics of the new level of the hierarchy of schemas. *Repetition of addition produces an unexpected whole which is equal to the sum of its parts although each level has its own unique structure which is characterized by the number of sources beyond the reversibility and substitution that are produced at each level*[7]. Furthermore, the double duality of schemas in relation to dimensions allows the efficacious communication of representations downward despite halving data loss at each level. All this indicates that schemas have a very odd structure that has not been noticed before. Adjacent Schemas on either side of a target schema are conjuncted to build the intervening target schema. There is a pairing of schemas so that the most macro and the most micro are in each case complementarities of each other. So there is a special relation between pluriverse/facet, kosmos/monad, world/ pattern, domain/form, meta-system/system. The schemas form a ring that unexpectedly connects the facet with the pluriverse. When taken as a set, schemas have some strange characteristics that are difficult to explain by reduction to a mathematical structure. Templates of understanding are different from the dimensional structure of objects. Each schema carries with it peculiar characteristics which can be developed into a 'formalism' of its own. These 'formalisms' are usually developed within disciplines and there are not many that are discipline independent. However, some of these 'formalisms' have been explored in a previous series of papers which take each schema as a subject on its own and discusses its interaction with the whole set of other schemas[8].

A Fundamental Distinction: Logos/Physus

Once the hierarchy of schemas has been identified and its relation to the dimensional structure that is underpinned by the Pascal Triangle has been elucidated, then we can attempt to understand the context within which schemas exist. It is possible to create a formal representation of each schema and apply that to the domain of a discipline. But here it is more important to establish the context of the schemas themselves as we attempt to understand what they are by clarifying their differences from other related things. Therefore, we start with a fundamental distinction between Physus[9] (which is the same as physical phenomena) and Logos (which is the same as physical theory). This distinction is fundamental within the Metaphysical Era of our Western worldview. This is the era that we are in at the present time which superseded the Mythopoietic Era about the time of Thales and Anaximander. Since the inception of the Metaphysical Era, a fundamental distinction between things has

[6] Deleuze in Logic of Sense (1990; New York : Columbia University Press) tells us that this dual series is the hallmark of symbolic (cf Lacan) structures in general.

[7] Deleuze in Logic of Sense talks about these sources in terms of singularities out of which the structural series unfolds within a field.

[8] http://holonomic.net

[9] Also transliterated as *phusis*

been made, i.e. between things that are thoughts and speech on the one hand, i.e. logos, and between natural growing things, i.e. physus, on the other. Both of these terms, according to these distinctions, are dynamic and are involved in genetic unfolding. They become flattened into distinctions like mind/body, or idea/matter, and other similar distinctions within our tradition. We must go back to the original Greek distinctions because these are more complex and interesting than later distinctions. Things that display physus and logos are finite as opposed to infinite. Certain things, such as "ourselves," display a combination of both physus and logos. When we engage in science we have a logos about the physus and in our experiments we have a physus specifically related to a logos (theory). So as *science* is manifested it produces theories about observations of experimental results from what Bacon called the torture of nature. Logos can break free of the physus and build all sorts of castles in the air and physus, as emergent phenomena, can break free of logos without corresponding to meaningful speeches at all. Most of human history has been a confrontation with this disharmony between physus and logos. But slowly mankind has learned to focus on the correspondence and coherence between physus and logos. Now, for Systems Engineers, this problem presents itself as the need for lots of documentation, as well as the necessity of dealing with the problem of the relation of the documentation to the things that are being built. We do specific audits at the end of our development cycle to make sure that the physically built design corresponds to the designed documentation, and we also make sure that the built system's functionality matches that which has been specified. *This possibility of a split between physus and logos haunts every system we build*. The development process is dynamic and the process of expressing both requirements and design in language are dynamic. Keeping these two dynamisms in lock step can be a major challenge and this is the challenge that is assigned to Systems Engineers who must maintain coordination and coherence between requirements, design and implementation. *So this distinction, between Physus and Logos that was defined in the ancient history of our tradition is still very important to us today in Systems Engineering. It is not just an arbitrary philosophical distinction, but one that we confront the reality of every day*. Systems are also finite, and they need to be built with a finite reserve of resources. But there are an infinite number of possible designs, infinite ways to fit things together, and infinite ways to implement and test that implementation. So the finitude and infinitude distinction is also important to us, and this underlies the logos/physus distinction.

The Ontic Hierarchy

The hierarchy of schemas is emergent, i.e. each one has its own unique characteristics that are non-reducible to the others and there is a relation of supervenence between the various ontological emergent hierarchical levels of the schemas. But the ontological hierarchy, so called because it is a projection of Being onto things that otherwise would merely exist, is not the same as another type of hierarchy called the ontic hierarchy. The ontic hierarchy is what cannot be reduced ultimately by scientific analysis to other things. An example of such a hierarchy might be gaia(?), social, organism, organ, cell, molecule, atom, particle, quark, string(?) or whatever thresholds of phenomena that you may subscribe to and designate as real. While the ontic hierarchy is created by the pressure of reduction, the ontological hierarchy is created by the pressure of skepticism. In other words we can be skeptical about whether a particular schema really exists or not and attempt to reduce it to other schemas. So, what ultimately stands up to scepticism is the ontological hierarchy of schemas, and what

ultimately stands up to reduction is the ontic hierarchy of the emergent levels of the organization of phenomena. Science discovers the *ontic* thresholds by projecting the *ontological* thresholds. As Systems Engineers we are dependent on the thresholds of phenomena and how they are described by science. We do not go out and invent our own thresholds of ontic phenomena. But we should not get the idea that what we do as Systems Engineers is not science. Science, as a discipline, operates precisely the same way as Systems Engineering and we should consider Systems Engineering as a kind of *design science*. We should study what the Philosophy of Science tells us about the way that science truly operates, rather than accepting the myths about it that have been created over the centuries. The concepts of those such as Popper, Lakatos, and Feyerabend can show us how it actually operates, and we will find that the way Science operates is very similar to the practice of Systems and Software Engineering. One of the key things that Popper pointed out was the importance of refutation. Any theory that is not refutable is actually philosophy. As Peter Naur[31] says: Designs are essentially theories. So testability is very important for design theories, as most Systems Engineers know. Kuhn taught us about Paradigm changes, and how our theories can be revolutionarily changed by an alteration of their fundamental assumptions. Paradigm changes have a big effect on designs such as when we attempt to implement object oriented designs rather than the more traditional functional designs, although Systems Engineering has not completely transitioned across this paradigm shift that alters all the various aspects of our designs as we attempt to apply new object oriented paradigms. With a viewpoint similar to Popper, Lakatos focused not on refutations, but saw scientific theories as conjectures which the scientist, working as part of a team, attempted to prove. The group would develop a research program, which they would pursue, based on their own self-definitions of the cutting edge of their discipline. Similarly, Systems Engineers, at times, attempt to push the envelope of technology while working in teams that define for themselves the ground rules of their projects. Feyerabend attempted to extend the work of Lakatos and drew negative conclusions about the usefulness of methods. His maxim was that one should use any means that works, and so one cannot dismiss out of hand or even crank approaches to problems as they might lead to something that works, which normative science could miss. All the different approaches to software should show us that there is a proliferation of very different ways to produce software implementations and a similar intrinsic diversity is also true of Systems Engineering. Philosophy of science has many things to tell us about the actual practice of science which was not understood until recently -- even by the scientists themselves who worked unselfconsciously on their problems without considering how they reached their results. Systems Engineering needs to use those unexpected results in order to frame its own projects within the scientific and technological domains. Systems Engineering and Science are interdependent. Scientists depend on large engineered instruments. Engineers depend on what scientists discover about ontic phenomena while experimenting with those instruments. There is no master/slave relationship between science and engineering. Engineering is just as important to science as science is to engineering. The practitioners of these disciplines are colleagues who need to mutually respect the contributions of the other. Systems Engineers need to be concerned with methods too, just as Software Engineers are concerned with methods. Software methods may be a subset of Systems Engineering Methods, but the two sets are not equal. *Systems Engineers cannot just use software methods without revising them for their*

different purposes and one of the main reasons for this is that Systems Engineering must combine the results of Software Engineering as well as other disciplines.

In some ways we can think of Systems Engineers as the opposite of Scientists in as much as that they are concerned with the synthesis of technological artifacts and not with the reduction of nature. This emphasis on synthesis is a key aspect of Systems Engineering because their work involves producing emergent effects in systems that are: *wholes greater than the sum of their parts*, i.e. gestalt systems, within contexts that are *wholes less than the sum of their parts*, i.e. proto-gestalt[10] meta-systems[11]. In this way the Systems Engineer is the latest addition to the tradition of craftsman, which is a tradition that is far older than that of scientist. With industrialization, the tradition of the craftsman was transformed into one of engineering. It is a peculiarity of the Western tradition that we are able to synthesize many different technological systems together, and this synthesis of different crafts and types of technology into even more comprehensive integrated wholes is what has necessitated the position of the Systems Engineer within aerospace and other industries. The Systems Engineer is the one who takes responsibility for the successful implementation of the whole system in its intended environment. In this process the Systems Engineer creates his own ontic hierarchy of emergent wholes with different characteristics composed of designed components. But, whatever the characters of these ontic components are, they must adhere to the template of schematic projection. Schematic projection is the underlying foundation because everything that is produced has some sort of spacetime envelope which is dimensional. These spacetime envelopes have their own schematic properties. The process of projection is itself temporal so schemas exist in both time and space and are, in fact, expressions of spacetime intervals of the sort talked about by relativity theory. In this sense schemas are not just static envelopes or templates of understanding, but are indeed processes which envelop everything that is projected into Being. Recently Peter Lynds[12] has discussed the fact that there is no determinate position with respect to time, and it is this nature of the interval in spactime that offers a different way of looking at Zeno's paradox. This is just a way of saying that there is a difference between Pure Being and Process Being and that we can never actually capture anything indeterminate and continuous as Pure Being, but that everything has only a Process Being which is probabilistic and indeterminate within a spacetime interval. This must affect the nature of the schema since we tend to reify each schema which is a projection that is actually probabilistic and indeterminate. All the syntheses which are projected by craftsmen, engineers, and systems engineers, end up being expressed as schemas of one kind or another – or as spacetime intervals. Prior to Lynd, William James called this the *specious present*, while others such as G.H. Mead talked about the fact that *it takes time for anything to become itself*. So all schemas are basically spacetime intervals of different dimensionalities, but these intervals also have an aspect that makes them templates of understanding because they are the grounds for the reference to the thing as well as the basis for the determination of its kind of essence, its idiosyncratic characteristics, *and* our interpretation of it as it is expressed in the gloss in language. The templates of understanding might be seen as the proto-synthesis on which the final synthesis of the artifact is based which, in turn, makes it an

[10] The proto-gestalt is the tacit knowledge of the implicate order of the gestalts.

[11] Open-scape

[12] Lynds, Peter (2003) "Zeno's Paradoxes: A Timely Solution" See http://philsci-archive.pitt.edu/archive/00001197/

emergent whole. Thus it behoves us as Systems Engineers to study these templates of understanding (or proto-syntheses) in order to further ground and synthesize our work with the projections that are the lifeblood of our own perspective on all the things in our world.

The Nonduals Between The Duals

We have now understood that Logos is related to the Ontological hierarchy of Schemas distilled by our scepticism and Physus is related to the Ontic Hierarchy of non-reducible things. Systems Engineers attempt to make emergent wholes that are gestalts (i.e. wholes greater than the sum of their parts) with emergent properties through the interaction of the natural wholes and the schemas. Artificial emergent wholes produced by Systems Engineers are virtual in the sense that they are possibilities that are not realized in nature, instead they are realized beyond nature in the artificial realm that we synthesize using what we have learned from science through a kind of tinkering that is the hallmark of all engineering practice. But if this tinkering is to be guided, then systems engineers must, as in science, recognize the non-dual realm of Order (i.e.Nomos) between Logos and Physus in the same way that scientists do. To paraphrase Einstein: the mysterious miracle is that mathematics can connect theory to the observations of experiments on nature. Math is the secret bridge between theory and practice in both science and engineering. When we speak of math we mean all the mathematical categories and the most basic of these is the Set. However, the mathematical foundation for our scientific and engineering work is lopsided and flawed in a way that is not fully recognized because the complement to the Set category is not represented in our mathematics. We have now discovered from studying other cultures such as India and China, that they had a different basis for their math and logic, which was the Mass (rather than the set). A *mass* is the complementary opposite of the *set*. The mass is a large body of identical instances that together produces macro effects through their micro interactions. A set, on the other hand, is a series of different elements called particulars, each of which is a different bundle of properties. A set cannot have more than one of the same kind of thing. Sets operate in reference to the differences between kinds. Thus the whole emphasis of the set is on the essences of the different things that make up the set. If you want to have repetitions of elements of the same kind then you must have a *bag*, and if it is ordered, then that is a *list*. The set emphasizes the difference of its elements and it is the natural complement of the mass, which emphasizes the identity of its elements. There is currently no mathematics of masses. Masses are forgotten in our tradition, even though our language has ways of talking about them, such as when we talk about a blade of grass in a yard. Grass is a mass and the blade is a counter of the instance of a leaf of grass that makes up the mass of the grass in the yard. The yard signifies the boundary of the mass of grass. The key point about mass is that it has emergent properties at the level equivalent to the set. On the other hand, although the set has no emergent properties, it does have particulars that have emergent properties and instances. Because of the emphasis on *sets* in our mathematics, we have a tendency towards analysis and reductionism, and, as a result, we tend to be sceptical about emergence. A shift toward considering the emergent properties is very important. For example, thermodynamics is a mass oriented science, while particle physics is set oriented. Until the recently discovered concept of negative entropy was accepted, thermodynamic systems were relegated to the backwater of scientific study. It is important to see how the set-like properties were segregated in physics from the mass-like properties of thermodynamics.

Let us think of a kind of mathematics that balances set-like and mass-like approaches that could be applied to Systems Engineering. We always design in a set-like manner, but when the system executes, or operates, then we are suddenly transferred into a mass-like behaviour as the various parts of the system are instantiated and begin interacting. Because, as Systems Engineers, we deal with emergent effects *at the macro scale,* which we try to initiate from designed micro components, we need a language to talk about this transition from design to execution (or operation). Many unexpected things happen in the realm of execution and operation and the mass-like properties exhibited there are not always what we intended or planned. For instance, we design a car, but when it goes out on the street it enters the mass of traffic so, consequently we need to see what the emergent attributes of traffic are and use that as a means to improve the set-like design of cars. The point is that while sets have a syllogistic logic related to universals, masses have a pervasion logic related to boundaries. We can reason about both sets and masses, but we have to use their natural logics, We cannot use set logic to think about masses and vice versa. In Systems Engineering we are continually dealing with sets and masses and their combinations. For instance, a combination of masses is a solution. Those are interpenetrated masses. Solutions (which combine masses) may have different properties than the masses on their own. Masses are unordered and follow the dictates of thermodynamics for the most part[13]. Local interactions between instances in a mass produce the emergent properties of the *mass as a whole* and they can be very different in various cases. For instance, the space of geometry is a mass of dimensionless points. Ideally we project coordinate grids on these masses. But from the viewpoint of physics, we know there is the *guage* phenomenon, which does not allow the external projection of coordinates. In spacetime there are geodesics, i.e. internal coordinates to the worldline of the particle moving though space. That is how the particle can appear to be in flat spaces along its route as it is actually moving through globally curved space. The gauge phenomenon is general, in masses there is no external point of view from which to project a coordinate set. Another point is the *Bekenstein Entropy Bound* and the *Holographic Principle*, which states that the entropy of something is one quarter of its surface area. That means that whatever is going on in a space can be written on the bound of the space. *This is a very profound principle that has many implications for schemas theory, due to the fact that each lower level schema is a surface for the next higher-level schema in the dimensional framework of Pascal's triangle*. There is information loss as we go towards the zero dimension while representing higher dimensions. It turns out that the schema differences appear as twin dimensional-jumps, and that is precisely a quartering of the information. This is because there are two dimensions per schema and two schemas per dimension. So in order to force a schema change it is necessary to descend by two dimensional-jumps. This causes a quartering of the higher schema to produce the lower schema. So, for instance, a system has only one quarter of the information of a meta-system. The other three quarters of lost information is spread between the anti-system, the entropy[14], and the final part of the information, which is not merely disordered, but actually destroyed[15]. This blocks our access to the higher-level emergence of the higher schema, which has become de-emergent in the representational production of the lower level schema. Reformatting is a

[13] Local negative entropy occurs sometimes, rather than just the universal of entropy.

[14] This, according to Bekenstein, is one quarter of the surface area of the meta-system.

[15] i.e., reformatted information

meta-system activity, which recycles lower level schemas to produce *other* lower level schemas. Entropy, on the other hand, is merely the introduction of disorder which counters the order of the higher level schema. So, in other words, there is a barrier between emergent levels of schemas, which cannot be breached merely by the assembly of lower level schemas. Thus, Bekenstein's Bound specifies precisely the emergent thresholds between the schemas. And that has to do with the loss of information to entropy as we move down to simpler and simpler representations and the reformatting or destruction of information. The opposite of this is the movement upward toward the n-dimensional, in which case we have *repetition* operating which repeats the information that is left, (although copying it is not enough to gain back the higher dimensional level). Rather, a negentropy from a singularity must reorder and reorganize the higher-level schema. This negative entropy (sometimes called Syntropy[16], or Ectropy[17]) reformats the information that is destroyed, disordered, and/or divided into the lower level schema and its anti-schema. In Systems Engineering we are continually fighting entropy. We are continually representing the systems that we wish to produce in requirements, and design or test documents. We know that the myriad of these documents do not completely capture the emergent whole that we are attempting to bring to manifestation within the world. It is human effort that bridges the gap to produce the allopoietic artificial systems that we attempt to produce. If it were not for our imaginations the theories through which these systems are formed would never manifest the emergent properties we intend. We are the singularities that move against entropy to produce emergence within our technical artifacts. Heidegger, in Being and Time[18] called that kind of singularity which we are: Dasein (being-in-the-world).

The Meta Level of Logos and Physus

Both Logos and Physus have meta-levels at which they interpenetrate each other. There is a *physus in the logos* and a *logos in the physus*. The physus in the logos is Logic. In other words, logic is the core of language, speech, and thought and it manifests as the logos arises from the physus. On the other hand, there is the logos in the physus which is the schema. That is because we project the schemas as a fundamental partitioning of things within our experience. This partitioning separates things from each other the way letters, phonemes, words, sentences, paragraphs, chapters, books, series of books, or libraries separate the parts of language, speech, and thought from each other. We lay schemas over physical things as they express their ontic natural complexes within spacetime. So Logic and Schemas are at the same level within our interpretive experience i.e., one applying to language as its essential core order, and the other applying to things as we experience them in their essential core order prior to kinds. Logic is also prior to kinds. Logic does not care what you are trying to say, but only how you relate the various propositions to each other. Our point is that logic can be syllogistic logic, or pervasion logic, although there may be other logics which bring our different fundamental categories, other than set and mass, into prominence. Logic is based fundamentally on three operators, *and*, *or*, and *not*, although there are many different kinds of logic. Some of those that we find significant are the Para-consistent[19] and Para-complete logics described by G. Priest. Along this vein are the

[16] Luigi Fantappié coined term in 1942 See http://www.sintropia.it/english/syntropy.htm

[17] See Edward Haskell (1972) Full Circle (New York: Gordon and Breach) http://www.synearth.net/Haskell/FC/FC.htm

[18] (1962; New York, Harper)

[19] http://plato.stanford.edu/entries/logic-paraconsistent/

Diamond Logic[20] of Hellerstein developed from G. Spencer Brown's Laws of Form[21] and the Matrics Logic[22] of August Stern. As Systems Engineers we have not begun in earnest to attempt to use formal methods as some of our Software colleagues have done. But in terms of our movement toward methods, and even toward formal methods, we need to keep our minds open to the importance of exotic logics. We need to be open to moving from the restricted economy of traditional 'first-order prepositional logic' to the general economy of many different kinds of logics for different purposes.

We also need to keep our minds open and allow for more aspects of Being than those recognized by standard logical formalisms. Being has four aspects: *presence*, *truth*, *reality*, and *identity*. Standard formal systems deal with presence, truth and identity *but not reality*. Thus there are only three standard properties for a standard formal system which are consistency, clarity (wellformedness) and completeness. However, if we add the aspect of Reality suddenly there are three other properties that are important, which are: verification, validation and coherence, and these make integration and synthesis possible. Notice that systems engineering differentiates validation from verification (did you build the right thing, and did you build it right). Also notice that as Systems Engineers we are particularly concerned with the coherence of interfaces as well as the system as a whole because it is coherence that allows the emergent properties to emerge as synergies. So, as Systems Engineers, we cannot *only* use standard formal logic, we must augment it with the scent of reality. It is interesting that the upshot of Model Theory[23] is that the addition of reality is what generates semantics or meaning. It is only when the emergent properties of the whole system arise that the system has meaning. Without the arising of emergent properties, the system is just so many pieces of things lying around uselessly on the ground, such as when we take apart a car. The meaning of the car is in its travel down the highway or roadways. When the emergent properties fall away so does the meaning of the thing within our world. Logic by itself is not enough. That is why I have proposed that we need to apply logic not only to truth, but to all the aspects of being, i.e. presence, identity, and reality as well. This is called Vajra Logic[24]. It is a logic which uses the form of the Toposes (the mathematical form of logics) with its binary characterization of statements to describe all the aspects of Being. If we add to that the capacity to deal with the paradox of Diamond Logic, and the ability to express both para-completeness and para-consistency (as does Matrix Logic) then we have a very strong logic which can deal with the contradictions that occur in the process of development of complex systems. Sometimes in systems, individual components need to be able to express divergent and even contradictory properties in order for the entire system to achieve synthesis. In the process of building the systems we also run into conundrums and enigmas that need to be comprehended. Normal logic does not fulfill these needs. As Systems Engineers we have to be open to appreciating and exploiting the characteristics of the more complex and exotic logics of both the syllogistic and pervasion types.

The logics express the different types of grammars or rules that can control speech or thought. The grammar of language is different from the pragmatics of thought or speech. One type deals with syntax and the other with semantics. When we turn to schemas we see

[20] (1997; Singapore ; New Jersey : World Scientific)
[21] (1969; Allen and Unwin, London)
[22] (1988; Amsterdam ; New York : North-Holland ; New York, N.Y., U.S.A.)
[23] Chang, C.C., Model Theory (1973; London: North-Holland)

[24] "Vajra Logic and Mathematical Meta-models for Meta-systems Engineering" INCOSE 2002 see http://holonomic.net

something similar. The schemas are dimensional articulations of the envelopes of spacetime as well as templates of understanding for things, i.e. the ontic physus. Both of these approaches to things are neglected in our tradition and that hinders their use by Systems Engineers in guiding their thought about design. *Systems Engineers must connect the software design to the hardware design, i.e. the dynamical information component to the matter energy component*. This connection between the two types of components is our way of projecting the physus/logos distinction into what we build. Software algorithms, which are represented by various software patterns and software language constructs, have a certain structure over and above the structure of logic. Hardware has its own structure, which is electrical, mechanical, etc. Producing systems where all these different kinds of objects can interface properly in order to allow their emergent properties to appear is very difficult. We do this by applying structures of logic and software languages and patterns. But also we must allow for the embodiment of the parts of the system as schematic envelopes in spacetime. Those envelopes were very static in the past, but as we become more sophisticated and allow for retooling and self-repair of systems these envelopes become more and more flexible. When we understand the intertwining of these envelopes through the schemas then we bring to bear some very robust resources because each schema has its own sets of formalisms (such as the kinds of logics that allow us to think about the design problems in new ways as we use the different schematic representations). The relationship of the schema to logic is through reference. When we say *this* or *that* we are pointing to a particular envelope in spacetime, a partitioning of spacetime within which something exists. First, we project (on it) the proto-synthesis of the schema, then we attempt to discover its essence so that we can use logical names to refer to these envelopes. The envelopes fit together like Russian dolls, each with its own proto-semantics and its own formalism expressing its properties. Also, schemas refer to logics because each formalism for a schema can have its Model Theory when we treat that schema as if it were a mathematical category. So logics, at some level, control the templates of understanding from within. Logics refer to schemas as a means of referring to things in spacetime. Those spacetime envelopes can be thought of as a *mass* of instances or as a *particular*, which is part of a *set*. Sets are arbitrary, while masses are non-arbitrary (even if they are random) because masses have emergent properties which are derived from the mass action of all their instances such as the waves on the sea. On the other hand, sets have no emergent properties and are just a pure collection of the different kinds they encapsulate. So a mass approach through pervasion logic is a more natural way to think of spacetime envelopes rather than the set approach which is seen though syllogistic logic which projects universals instead of boundaries and sees different essences rather than similarities. So we really need the power of the mass approach to things in order to realize the full power of schemas where we identify a dimensional spacetime envelope and see it as pervaded by the proto-synthesis of the schema prior to its pervasion by its kind (called a Form by Plato because he conflated the schema form with its essence). The patterning of the schema itself is the proto-essence of every schematic partition. On that proto-essence the *kindness* of the actual essence of the thing is built for each spacetime partition. These spacetime partitions may be collected into an empty set, thus filling it with different kinds of things. Although sets without particulars are empty, a mass is never really empty because it must have its instances to exhibit emergent properties. Yes, we can project a de-emergent null mass without

instances and consider infinite null masses to be what Democritus and the Taoists described as the Void, just as an infinite extent of empty sets[25] could be what the Buddhists called Emptiness. This compensatory pattern of the difference between sets and masses can also be seen in the isomorphism between the two logics of syllogism and pervasion. Peirce made the point that the three statements of the syllogism can be arranged in three different ways to produce induction, deduction, and abduction (hypothesis forming from cases). Similarly there is structure that is similar for pervasion logic that reasons about boundaries rather than universals. The key question for pervasion logics is whether we are inside or outside a boundary and thus within the mass or outside the mass. If we are inside the mass we are pervaded by its properties. It has been noted that Plato's idea of forms was probably mass-like originally. That is to say that Beauty is a mass, and that all beautiful things are seen as instances of beauty. If it is within the bound of the beautiful, then it is pervaded by beauty and has the emergent properties of the beautiful which have to do with harmony and proportion. This approach obviates the need for a place for the transcendental realm of the source forms of Plato. The Mass of the Beautiful is simply all the beautiful things, meaning that mass has its own *sui generis* properties that are beyond those of the beautiful things themselves. For instance, all the beauties of nature have a profound effect on the soul. It was Aristotle that broke with his predecessors and established the set-like bias of our tradition[26].

It is a similar story with adding the reality aspect to presence, identity and truth. This relates to the schemas because schemas are about the minimal representation of things prior to determining their kindness. Because they are about things there is some measure of reality involved in the identification of the schemas that goes beyond pure logical formalism. Before real objects of experience are schematized we know them as spacetime envelopes. Those envelopes have their own structure that is different for each envelope type. That structure is intelligible even without knowing the kinds of the things that are taking that spacetime configuration. That intelligibility is a sort of infrastructure that all things of that type share. If we understand those infrastructures and how we fit together we have a better chance of designing artificial sets of spacetime envelopes that fit together well. Reality brings with it the ability to verify, to validate, and to discover the coherence that allows system integration. Formal systems by themselves with just presence, identity and truth do not allow for these properties and thus remain disconnected from reality. Thus it is good news for us that Being encompasses not just presence, identity, and truth but also reality. Systems Engineers need to deal with *reality* every day and to reason about reality in relation to the other aspects of Being. It is that *realism* of the systems engineer that brings him to write "The Unwritten Laws of Systems Engineering.[27]"

It is clear that it is schemas that bring with them the need to expand the aspects of Being as well as the kinds of logic that are acceptable. We especially need a logic like that of Hellerstein which attacks paradoxes and allows us to frame anamorphs[28] that solve paradoxes. But that logic must be a Vajra Logic which applies to all the aspects of Being equally, rather than only focusing on Truth. Statements are not just true and untrue; they can be real or illusory as well. The Systems Engineer must deal with all the aspects of

[25] See http://emptysets.com Kajetan Guz

[26] The Discovery of Things: Aristotle's Categories and Their Context by Wolfgang-Rainer Mann (2000; Princeton, N.J. : Princeton UP)

[27] David F. McClinton INCOSE 1994; A commentary on this classic paper is given in the full version of this paper at http://holonomic.net

[28] See Donald Kunze on "Boundary Logic" at http://art3idea.ce.psu.edu/boundaries/mainpage/directory.html

Being equally. He deals with what is present and absent, what is identical and different, what is real and unreal, and what is true and untrue every day while attempting to be just and practical at the same time.

How Logic Relates To Schemas

There are three different terms at the meta-level above logos and physus which are Logic, Schemas, and Mathesis. Logic relates to Schemas in terms of the Philosophical Categories, i.e. the highest concepts that connect pure ideas to things. These are concepts like quality/quantity, causality, part/whole, etc. and these were identified by Arsitotle as the most general statements that can be made about any substance or kind of thing. They were also identified by Kant in his table of categories. Kant goes on to identify the schemas as being related to each dialectical set of categories in his table. For our part we prefer the categorical scheme developed by Ingvar Johansson[29] which is built on the work of Husserl. There are myriad categorical schemes available from the history of our philosophical tradition. Schemas are things that assume causal relations in time, that combine quality and quantity, that have part/whole relations, etc. We use our categories to think about things in their most basic forms. It is as if the philosophical categories were the infrastructure needed to create the formalisms that describe each schema. The same categories show up in each formalism in different ways that are suitable to that schema. How these philosophical categories operate over the schemas and are manipulated by logic would be a study in itself. Here it is only necessary to mention that there is this high level connection between the schemas and logic via the philosophical categories. This connection is what we use to create the naïve view of the world. Durkheim said that the Kantian Categories are social, so if we consider them as social constructs rather than universals of the mind, then we can consider the cultural determinateness of the categories as well as logics and the schemas. It is interesting to note that in our culture and in an idealist tradition schemas are not well described and the mass-like way of looking at things is also not well developed. As a result meaning and reality are not accounted for. It is the pragmatism of C.S. Peirce in his First Category that comes closest to giving a grounding to Schemas theory. Peirce names three categories. These are called First, Second and Third. The First is the Isolated Thing that shows up[30] without relation to anything else. Second are Relations. Third are Continuities. To this we add a Fourth category from B. Fuller which is called Synergies. To all of this we add Zeroths which is the background out of which the Firsts appear. What we first notice is that the differences between these categories are the kinds of Being. *But beyond that we notice that each schema is in fact an articulation of all five of these categories.* Every schema takes the lower level schemas as firsts. It relates those lower level schemas to each other and then produces a continuity which is the emergent characteristics of the schema above the discontinuities of the lower level schema. Each schema also has its inner coherence, which is a synergy at its own level. The discontinuities between the various schemas are expressions of the Zeroth category. If we say that each schema is an expression of the Peircian categories then it must also be an expression of all the kinds of Being. Thus what we call a face of the world is a synthesis of the various fragments of Being.

How Schemas Relates To Mathesis

Schemas also relate to Mathesis, which is the faculty for the production of order

[29] See http://hem.passagen.se/ijohansson/

[30] Like “hyle” (content, matter) in Husserl

(nomos). The schemas relate to Mathesis through representations. Our representation that uses Pascal's triangle as a way to understand the nesting of the different schemas is a case in point. Representations simplify. So we get representations as we move toward zero dimension *down* the scale of schemas. Moving *up* (as we said before) is what Deleuze refers to as Repetition. We are familiar with representation but not repetition. Repetition produces more and more complex items. We are instead geared toward reduction and simplification. However, as we are ecstatically projecting Being, there is a dimensional overflowing and a natural repetition that we are engaged in that we suppress in favor of an emphasis on representation alone. As templates of understanding, the schemas serve as sites for both representation and repetition. We can represent the class of envelopes of spacetime entities via the formalism associated with a schema. But every application to a new kind of thing, or an instance of that kind, is a repetition that reasserts the infrastructure or proto-synthesis that the schema represents as the foundations of understanding prior to the determination of kindness, or individual peculiarities, or interpretations. When you look at the world it is full of both representations and repetitions, but we only look at the representations and suppress the repetitions -- such as the way that we repress the reality and mass approaches to things. Schema Theory breaks that impasse and appeals to all three suppressed elements to ground our understanding of Schemas Theory. *We need repetition because otherwise we cannot travel both up and down Pascal's triangle*. We have already described why we need reality and mass approaches to things. As systems engineers we are stuck with many representations. And we seem to repeat those representations endlessly. Now we are even asked by CMMI to "plan our planning" and "monitor our monitoring" and "configuration manage our configuration management." In other words we are called upon to make meta-level representations and to repeat those! All those repetitions of documents cannot capture the system that is being built in full[31]. *That is because there is a dimensional difference between the system and the documentation. The documentation must deal with an essential information loss due to de-emergence. It is operating in an arena in which entropy must be confronted as the enemy of the implementation at every turn*. It is strange is that although there is no amount of repetition of representations that will capture the as-built system; it is also true that the system as a singularity arises out of the field of those repeated representations under the right conditions. *The whole question becomes: How do we harness the negative entropy of the human element in project development as we bring about the necessary order that must be unfolded for the emergent properties to appear as intended*? Humans wander about in the trash heap of the repeated representations and somehow manage to bring the system together in spite of the entropy they are fighting against. This would not be possible if there were not some ultra-efficacies at work. One of those ultra-efficacies are the schemas themselves. They allow communication between representations at various levels. They allow the intertransformation of information between representations at the same level. When you look at it deeply you see that the schemas are the backbone on which the flesh of every system is hung. It is the ultra-efficacy of our joint projection of the schemas that allow us to work together on the same system. In other words, the schemas are an intersubjective social invention and construction (or projection) that we share in common and this

[31] Naur, Peter, "Programming as theory building," Microprocessing and Microprogramming, 15: 253-261, 1985, reprinted in Computing: A Human Activity, (1992; NY: Addison-Wesley, ACM Press, pp. 37-49)

inhabits our mutual, conversational, team memory. And I would like to venture that this projection is housed in a communal intermediate memory that is purely social and stands between long-term memory and short-term memory. We store our conversation trees in this collective memory and it is those trees that are structured by the schemas, because that is what allows us to mutually refer to the same spacetime envelope. The schemas are not just individual projections but group projections. They are, in fact, what allows our conversation trees to interface with the world which portions spacetime into things. Conversation occurs as we wander though the world. As we wander we point to this or that and indicate a schema even though we do not yet know what it is that we are pointing at, (and this is because it has not been completely designed yet). The theory of the design is the gloss on the conversation tree that gives rise to the mutually held theory. We cannot capture the theory because it exists in a communal memory which we have an imperfect access to if we are not in conversation with the others with whom we are doing the design work.

How Logic Relates To Mathesis

Logic relates to Mathesis via Model Theory. Model Theory has to do with the statements one may make about a mathematical category. But once we have a full meta-Model Theory then we can see that schemas connect to mathematical categories that then connect to theories via models. This is the arc of science, which is different from the arc of philosophy, which directly connects the schemas to logic. This indirect connection is powerful because the inherent order of Mathematics can be a useful guide for researching the type of Math that underlies the actions of particular phenomena. Model Theory is not only very important, but is more straightforward in our tradition than representation theory. Logic and Math are well developed in our tradition and connecting them via models is a fairly straightforward way of analyzing a problem. So there is actually little to say about Model Theory except that we must extend it to deal with reality rather than just truth, presence and identity. Once we include reality and apply Vajra logic (plus other exotic logics), then, Model Theory, as it is established, will serve us well. In Systems Engineering this shows up as *formal methods* that *combine mathematical structures* with *logical structures*.

References

Palmer, Kent D., "General Schemas Theory", CSER 2004; See http://holonomic.net

Palmer, Kent D., "A Framework for Exploring General Systems Theory"; See http://holonomic.net

Biography

Kent D. Palmer Ph.D. works as a Systems Engineer at a major aerospace company engaged in CMMI Systems Engineering process improvement and Systems Engineering Practice. He is also working toward a second (part-time, external) Ph.D. in Systems Engineering at the Systems Engineering and Evaluation Center (SEEC) which is part of the University of South Australia. See http://archonic.net and http://holonomic.net.