# OO "software object" as a living object

Albertina Lourenci
Post-doctorate fellow. Laboratory of Integrated Systems-
Department of Electronic Systems Engineering
Polytechnic School University of São Paulo
Avenida Professor Luciano Gualberto, 158 Travessa 3 Butantã São Paulo
e-mail:Lourenci@lsi.usp.br

**Abstract.** The language game works in symbiosis with the human brain .It is not enough to simulate semiosis in the sense of conceiving a minimal structure of signification able to unfold a generative trajectory of the OO alive software object consisting of deep and surface levels. **Agile software developers** have been paving the way to this significant turn to build the software object as a whole alive unit. First they criticize software engineering as a metaphor. Second the search for the right one. This entails to take into consideration the nature of the software object and how it unfolds. A living software object must necessarily reflect the interaction of human beings with the real world as well as how the world is as a mirror. An ecodesign model to design and plan sustainable cities being implemented in the prototype-based OO programming language Self is given. These cities when built behave as true living organisms .

## 1. Introduction

To cope with the challenges posed by the complexity of our daily lives, breakthroughs in computer science enabling us to deal with the software object as a living organism has become a must!

To date the most striking thing pervading all the diversified trends in software development is the attempt to remove the hurdle that has been hindering the achievement of this goal: the questioning of software engineering as a metaphor. It is consensus that according to Djikstra the software engineering's subject matter is located solely in the Machine no longer holds: *Our task is only to build the (formal) Machine to satisfy a given (formal) specification at interface a; this specification is a logical firewall (Lourenci, 2001).*

Lehman and Ramil (Lehman and Ramil, 2002) classify programs that follow this statement as S-types (S-specification) and propose E-types (E-evolution) : *an E-type program is one whose acceptability depends on the perception, judgement and degree of satisfaction of appropriate stakeholder (s). Software used to solve a problem or address an application in a real world domain is in general of this type. The system is intrinsically evolutionary. Its relationship to the real world may be described as model-like reflection. Properties such as these make implementation and use of the systems a learning experience.*

Curiously they do not consider OO programs as E-types! And are not aware of the recent OO concerns with software evolution.

Kristen Nygaard conceived OO paradigm in late forties before the advent of the computer exactly to simulate phenomena of the real world. In the nineties trends such as design patterns, aspects and reflective architectures and prototypes together with the alternative mechanism of inheritance called delegation have been introduced to deal with "software evolution" (among many others) . OO programs have followed the S-types trends obviously! More and more the importance of delegation has been becoming clearer and it has been simulated in composition filters, in Lava (Java with delegation) (Kniesel, 2000), etc. Yet this term has become not evolutionary enough and the more radical "unanticipated software evolution" has been introduced especially by agile software developers. Of course, through trial and error life is being infused into software.

What does evolution mean? Simply put, the magic word is **double articulation: the ability to create out of finite means something seemingly infinite**. Fortunately it pervades life and language as information-carrying and goal directed systems. Hence inspiration can be drawn from nature and culture.

On the one hand, it means to realize that the genetic code is responsible for the myriad of manifestations of life. Accurate replication of genomic DNA is essential to the lives of all cells and organisms. Each time a cell divides, its entire genome must be duplicated and complex enzymatic machinery is required to copy the large DNA molecules that make up both prokaryotic and eukaryotic chromosomes. Hence energy enters in! Moreover, cells have evolved mechanisms to correct mistakes that sometimes occur during DNA replication and to repair DNA damage that can result from the action of environmental agents, such as radiation. Despite the importance of accurate DNA replication and maintenance, all genomes are far from static.. They yearn for "unanticipated software evolution". Mutations and gene rearrangements are needed to maintain genetic variation between individuals. Recombination between homologous chromosomes during meiosis plays an important role in this process by allowing parental genes to be rearranged into new combinations in the next generation. Hence a careful balance between maintenance and variation of genetic information is critical to the development of individual organisms as well as to evolution of the species (Cooper:175, 1997). Of course, the genetic code generates the myriad of forms of life on Mother Earth.

Emmeche and Hoffmeyer pave the way gracefully to show life as learning and thought, as memory systems, as cognitive systems, as computers and finally as linguistic and semiotic systems (Emmeche and Hoffmeyer, 1991).

Indeed Winfried Nöth (Nöth, 2002a) blurs the difference between allopoietic and autopoietic systems when he views them as semiotic machines. The former, destitute of all originality , of all initiative and the latter, able of self-reference, autonomy, self-control and self-maintenance and reproduction. On the one hand, Peirce's concepts of semiosis and quasi-semiosis tune with Prigogine's self-organizing theory: self-organization pervades mineral and organic worlds (Prigogine, I. and Stengers, I.). On the other hand, doubts concerning the genuine autonomy of human consciousness have been raised. Cultura and genetic factors influence human decisions hindering the expression of really free will. I will not delve deeper here because Nöth bases his reasoning on Peircean semiosis. I advance one that promotes its full understanding due to be grounded directly on meaning.

On the other hand, the semiotician Roman Jakobson (Jakobson,, 1973) compares: *Among all information-carrying systems the genetic code is the only one which shares with the verbal code a sequential arrangement of discrete subunits; phonemes in language and nucleotides…in the genetic code.*

Moreover Marcel Florkin (Florkin, 1974) places the accent on the biosemiotic aspects of molecular evolution , i.e., the intensive aspects of information involved in the impact of natural selection acting at the level of the organism: *biomolecular order is governed by systems of signification which we may consider in a structuralist (intensive) perspective besides the thermodynamic viewpoint and the quantifying (extensive) viewpoint of the information theory. (Florkin 1974:13).*

He terms the mimimal configurational aspects of molecular signification *biosemes* (Emmeche and Hoffmeyer, 1991).

Likewise the semiotician Greimas in Structural Semantics (Greimas 1983) modeled his minimal unit of signification "the seme" on the distinctive features of the phoneme of Roman Jakobson who founded the Prague School, which invented and developed phonology between 1929 and 1939 as opposed to the pheme. Each phoneme is a bundle of phemes or a bundle of distinctive features such as – voiced and + voiced in the opposition of the phonemes /d/ vs /t/ in English that only exist in a structure. This binary opposition also happens at the level of the DNA, Pyrimidines such as guanine base-pairs cytosyne and the purines such as adenine base-pairs thymine or uracyl.

Likewise the oppositions (white vs black) that allow the formation of semantic axes between the semes not only do represent the specific difference between the two terms but also what is semantically common to the pair of concepts or what articulates the opposition. The opposition between the two terms may only be considered elementary because there is the absence of colour as the semic category, uniting semantically the concepts black and white. Hence the binary semantic structure has two semantic sides: one that consists in the opposition and one that consists of the common element of the opposition. The oppositive element founds a disjunction relation and the semic category, common to both elements a conjonction relation. Disjonction and conjonction belong to the semic category of jonction. This is the double nature of his minimalist definition of the elementary structure of signification. The model for its instruction is the semantic axis, The terminal points of the axis represent the elementary oppositive concepts, defined as semes. The axis that unites both semes, symbolizes the common semantic element: the so-called semic category (Nöth, 2002b).

Indeed just one semic category is enough to order and produce due to successive investments at each generative instance a micro-univers of discourse. This will become clearer later.
Curiously Richard Gabriel, another outstanding computer scientist that migrated to OO community, deeply interested in making us realize that programming is essential for everybody and that everyone can become a programmer, proposed a deconstruction exercise at the Feyerabend Workshop at the Sixth European Conference on Pattern Languages of Programs held at Irsee Germany from July 4-8 2001. The goals of the brainstorm were to uncover hidden assumptions and devise new ways to move forward. The participants were invited to think of dozens of pairs such as science/art, aligned/contradictory, independent/crosscutting, subject/object, reflective/opaque to try to articulate what is wrong with the status quo of software and find new solutions (Gabriel, 2002). He is also interested in systems as living organisms. Certainly if he were aware of Greimas's semiotic project he could have delved deeper into a powerful tool to apprehend the essence of the generative mechanisms of language such as the generative trajectory of discourse, where the elementary structure of signification introjects itself from deep level structures to discourse (Figure 1).

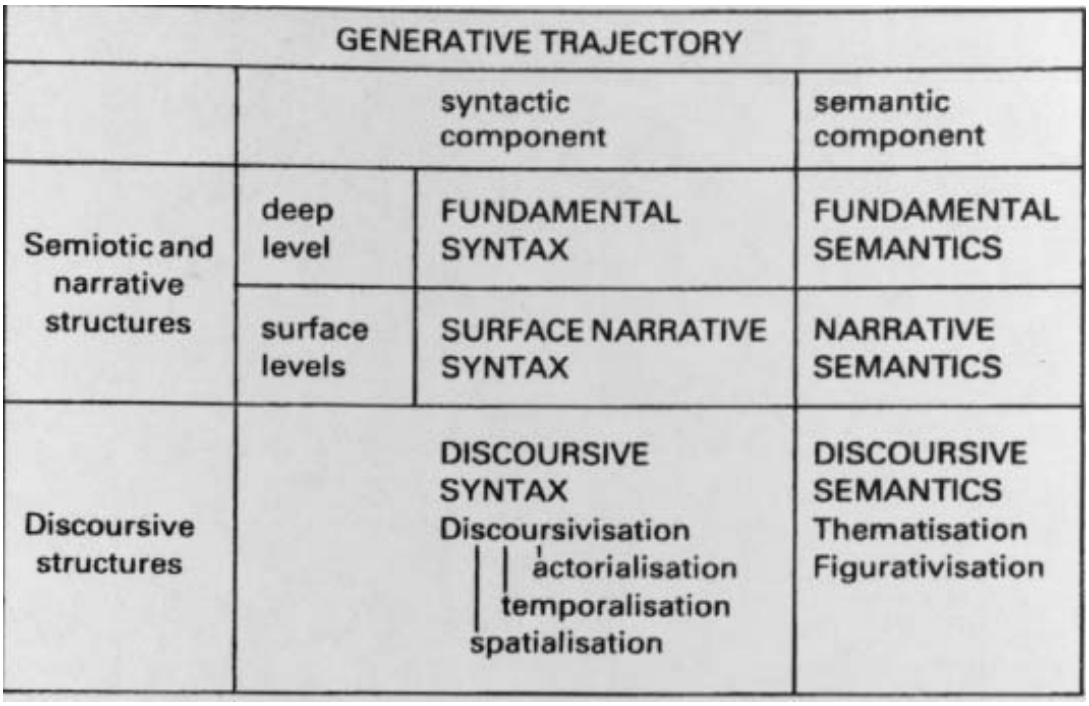



Figure 1.The generative trajectory of discourse. The elementary structure of meaning generates all levels of the generative trajectory of discourse. It can also be articulated as the semiotic square. It also unfolds all its levels.

Future research will exploit Greimas's project to throw light on the cloudiness of nomenclature linked to several approaches gathered under the umbrella of aspect-oriented programming, another promising technique to create living programs if aspects are clearly introduced at all levels of the software object and especially expressed as objects like in Us, the subjective version of Self (Smith and Ungar, 1996)... If we are successful in expliciting these basic generative mechanisms of modeling while isomorphic reasoning structures pervading all levels of the software object, this not only does enhance traceability and change but also opens the gate to introduce a process known as semiosis in the very heart of software.
In this sense, semiosis is a logic of becoming. The full understanding of its dynamics turns out to be no less than the definition of life (Sebeok, 1968)
Greimas's semiotic project encompasses concerns not only with cognitive aspects but also with sensible aspects of the language. He identifies feeling with the very principle of life (Greimas, 1991). In Figure 1, Greimas adds *in distinguishing between the different deep levels of semiotic structures in general, we can say that the deep structures are virtual, the semio-narrative structures actualized, and the discursive structures realizing. On the other hand, in designating the different phases of the modalisation of the acting subject*

*(sujet de faire) (of acquiring modal competence), we can divide the modalities into virtualities (wishing and needing to do (vouloir- et devoir-faire)) actualizations (being able and knowing how to do (pouvoir- et savoir-faire)) and realizations (making-to-be of doing (faire-être)(.* Greimas :1979:93-94.
This way he implodes the hierarchical way of seeing Western languages and a more pluridimensional, topological and relational structure unfolds, blurring the borders between prose and poetry. From a Greimasian viewpoint, semiosis is thus the travel of the minimal structure of signification along the generative trajectory of discourse.
Why have I chosen the DNA molecule and Greimas's semiotic project to introduce you to the conceptualization of the software object as living object? Why not Christopher Alexander or many others concerned with generativity such as René Thom, Mandelbrot, the physicist David Bohm (implicate order tuning with the hindu philosophy), Maturana and Varela, M.C. Escher, Prigogine?

First Alexander's influence is well known in the class and design pattern-based community and the current state of my research, the unfolding of "The Model of Primary, Secondary and Tertiary Waves to design and plan sustainable cities" is a fruit of my inspiration in my biological studies and the pungent effort of studying semiotics for being part of the background of an architect. Greimas also unfolded his idea into a Semiotic Dictionary in the style of Alexander entitled . Semiotics and Language: An analytical Dictionary (Greimas, A.J. and Court, J., 1982.
Second, painful or not they led me not only to the introduction of the Multi-Dimensional Separation of Concerns paradigm at the level of domain modeling. but also to the emergence of a new geometric consciousness tuning with the search for infinite manifested in the tilings from the Dutch artist M.C.Escher (Figure 10). All the above biological knowledge has been perfectly introjected thanks to the richness of the symmetry groups of the plane (crystallographic groups), similarity and conformal symmetry groups, which are also fractals). This enables composition of concerns neatly. And of course a new programming and software development consciousness. My ideas when implemented in a friendly software system have nothing to do with the machine-like way of aspect-oriented-programming though! My aim here is to propose to you that the way of dealing with the domain model may be reflected in the software architecture and in the code implementation level. It is essentially life and when my sustainable cities are built, they will work as autopoietic systems. Of course if and only if the OO software object mimics a semiotic object in the Greimasian sense! Us, the semiotic version of the prototype based programming language Self embellished with aspects and reflection may fulfill this goal due to its already highly interactive and collaborative nature due to Kansas, a multi-user programmable virtual reality where we can program side by side at long distances.

## 3. Postmodernism method

Of course poetry and life are mysteries that dovetail myriad of visions relentlessly. Only Postmodernist methods can cope with its exuberance.
Likewise my exposition holds out for building software free from the search for the ultimate foundations of everything. Especially when these are unattainable and in the endeavour to unravel them biased visions emerge, smashing true ones. Hence such a building would be a conversation from which no one is excluded and in which no-one holds a privileged position. Once of course the programmer is engaged in serving mankind above all. In hindering Mother Earth from collapse.
Its role purpose is to keep conversation going. In practical terms, it reflects the form of a revolt against the S-types expressed by diversified trends in the last fifteen years. It has "attacked the intellectual conditions that allow for and tolerate the dominance of heavy methodologies, that hinder me to implement my ecodesign model entitled The Model of Primary, Secondary and Tertiary Waves to design and plan sustainable cities. These cities when built shall behave as living organisms.
Postmodern philosophy has been powered by a simple question: On what basis can a claim be made for a privileged status of one theoretical viewpoint over another? How is language used to maintain the hegemony of the mainstream computer science instead of permitting an open dialogue? Why not to enhance inter-, multi- and transdisciplinary synergies? Only these mimic how life is! The effects of these questions and questioning of the Brown Agenda all over the world have been profoundly destabilizing and potentially anarchic, enhancing the Babel effect apparently (Dear, 2000). The fourth law of thermodynamics discovered

by the ecologist Joergensen in the early nineties (Joergensen, 1993) guarantees the complexification of the world despite the second law of thermodynamics that leads everything to death.
However what is life? It is becoming increasingly evident that without chaotic systems, without dissipative structures (Prigogine) there could be no order out of chaos, that is order of the self-organizing variety. And without order from chaos, neither life nor semiosis as we know them could sustain themselves, nor could they have emerged in the first place. It seems our postmodern world is favoring the emergence of these conditions. Kant says the easiest way to reach peace is through anarchy. Apparently there is no reason to be pessimistic …

## 4. How are recent trends in OO community favouring the appearance of the OO software object as an alive object?

Jim Coplien in a recent invited keynote address entitled *It's time to kill Software Engineering* at the 15th Brazilian Symposium on Software Engineering October 3 to 5, 2001 in Rio de Janeiro, Brazil endeavored to show the inadequacy of the term Software Engineering as a metaphor to grant a scientific status to computer science. He ridicules the attempts of S-type methods to convince us that artificial intelligence is intelligent[1].
I paraphrase what the semiotician Eric Landowski grasps of the gist of semiotics, replacing semiotician by programmer to express Coplien's search for a right metaphor for software: *the know-how of the "programmer" is concerned less with science than with craftsmanship and demands something like an art of writing.*
He proposes explicitly there should be something like a master of Fine Arts in Programming. Before likening the software object to the semiotic object especially conceived within the context of Greimas's semiotic project, I would like to express what art is according to the Nazi philosopher Martin Heidegger and how agile software development (Martin Fowler includes design pattern movement as agile!) enhances what and how art is! Of course I am an artist and art provokes insight, it transforms the one who is exposed to it! A far cry from the scientific discourse! Hence this approach intends to induce the reader to reach his/her own conclusions.
In the essay *Der Ursprung des Kunstwerkes in Holzwege* or The origin of the work-of-art, Heidegger tries to apprehend the thingness of the thing, the thingness of the product and the thingness of the work-of-art.[2]
However he fails in his search of the relationship between the thing and the work-of-art. Why, if this was very well evidenced in the poetry-thing from Rainer Maria Rilke and needless to say in the trends that have pervaded modern and contemporary art as a reaction to abstract art and its concern with the divine. He appreciated Rilke as the metaphysician and deemed him and Nietzche as the modern prophets! But this phase of his life he considered a dark period.
At the roots of this judgement was his affiliation to the Nazi Party which deemed Modern Art as degenerated. However this trend has grown taller, insofar as art itself has been considered a thing to which we simply make things, mimicking what we make with things. The climax of this movement is Robert Rauschenberg's art-thing. He has collected tires, cardboard, cans, old newspapers and all stuff good for recycling on the streets of New York to make his work of art and hence turning art a thing itself. What's the aim of this behaviour?
Nothing else than to evidence the way of being of art, what was fabulously exploited in the realm of hermeneutic philosophy. Yes, the way of being of art is hermeneutic or essentially it is a game theory! What does this reasoning have to do with software? Well, the essence of art is to create artifacts. It was exactly Rilke who put side by side his already known transformation of the exterior in interiority (*Innerlichkeit*) another opposite process through which the interiority by means of art is transformed into exterior, in thing (*Dinge*). He accomplished this in his famous poem *Der Panther* under the influence of the sculptor Rodin, who warned him he was unable to "see" . In this poem, the panther speaks out and the poet Rilke is dissolved!! When we model, this is the process that enable us to model. This is the essence of object-oriented programming. This is its great achievement compared to other styles. Or the succinct ability to put a "continent of knowledge" in the form of an "island". What's being questioned today is the inability of making the objects to speak by themselves as Rilke achieved!

---

[2] Object oriented programming is an attempt to talk about the world in a language of things = objects!

This begs a question: Does this mean that the great philosopher Heidegger add nothing to the understanding of art? Those who believe this are thoroughly mistaken. He fathomed the being of art! Due to lack of time I will not delve deeper into this. In sum, Heidegger is fortunate when he expresses that art is the unraveling of truth in a Gestalt (= image, thing, geometric figure, object, artifact, project, etc).

It is what is brought forth by creation as the disclosing of the emergence of the being (becoming). The truth hence emerges as a Poem. All art is essentially a Poem in the broad sense of the word. Hence Art reveals the truth as Poem. I hope this has triggered an insight in the reader's mind enabling him/her to understand the depth of Coplien's intuition about the necessary metaphor for the software object. S-types are based on old scientific methods linked to the Newtonian physics. One needs more qualitative reasoning! Like those from Human SCIENCES!

Heidegger also brings forward that the determination of the building of the being of the truth must be in alignment with the building of the being of the thing, the essence of the sentence and the truth! Here the Greimasian semiotics enters in to enable you to achieve this goal: an art of writing! To enable you to have a Master of Fine Arts in programming and software development!

And apply Literature as the right metaphor for software development. Indeed I am genotypically a physicist and surely I have become an architect because I spent my teens reading Sartre, Camus, Dostoievsky, Shakespeare, Kafka, Proust, Dickens, Balzac, the best of modern literature in Brazil.

This sounds too serious? Who said that art is serious? Above I was talking about philosophy and our duty towards Mother Earth and mankind! Art is playful, pure joy, a game of creation if you simply are open to it!

Hence Alistair Cockburn's the cooperative game called software development enable us to accomplish serious tasks playing or dialoguing:

*Software development is a cooperative game of invention and communication. It never was 'engineering", despite all the advertising to that effect. Software development consists of nothing but ideas, made concrete. It consists of people inventing and communicating, working through a problem they don't yet understand, and which keeps changing, creating a solution they don't yet understand and which keeps changing, expressing their ideas using very restricted languages that they scarcely understand, to an interpreter unforgiving of error.*

Alistair is our Robert Rauschenberg. (his work is known internationally). If Rauschenberg had had the chance to attend the invited talk-show entitled *People and the limits of methodology* at ECOOP'01 in Budapest, he would realize Alistair is one of his soul's clone especially tailored to act in software to make us dialogue with the things around us and perceive that the world around us is information! Peirce (1838-1914) uncovered this and founded semiotics! Post-quantum mechanics today is reaching the same conclusion! Science is evolving! Why does not computer science mimic it?

I believe it is not necessary to say more about Coplien's and Alistair's outstanding research.

However people steeped in the cherished belief that there is a ready-made, fixed-for-all-time world out there play havoc in favour of a processual view. *Fractals, chaos theory, Prigogine's physics of complexity spill over into vague notions of interconnected wholes, the interrelatedness of all things, their codependency, their collaboration in bringing forth what goes as the real world, and a dissolution of mind/body, subject/object, knower/known, continuity/discontinuity, and form/content dichotomies* (Merrell, 1999).

They enable us to see the universe whether we like it or not as cooperative, collective effort toward the goal of spreading out the heat in the most expedient way possible. Such pattern formation is what is meant by self-organization, although there is no fixed self! There is no stable, detached semiotic agent that is bringing about any organization. The organization just is! Just blooms! X-programmers bet on this and reminds us that this means constant learning and courage! The greatest virtue according to Socrates, insofar as without courage no other virtue is worth! Art is really playful? When you know how to play! Because the game in itself is serious! This is why the traditional programmers adhere to the heavy methodologies, because there they can manifest their egos irresponsibly without being concerned if programming is for everybody and that everybody needs to program to fulfill their tasks efficiently and synergetically to create a better world as soon as possible because Mother Earth cannot await to see the destruction of life on its surface!

Surely Agile Software Development triggers the necessary turn towards systems as living organisms. Moreover it enables us to give a special meaning to software architecture as the stage where the inventive game happens. This experience with the time being may be captured as a highly flexible and interactive prototype-based multi-agent architecture (Lourenci, 2002a, b). I mean it is possible to extend through Greimasian concepts the prototypes to cope with multi-agent properties easily.

## 5. The Model of Primary, Secondary and Tertiary Waves (MPSTW) to design and plan sustainable cities seen through Greimasian eyes

My Model of Primary, Secondary and Tertiary Wave to design and plan sustainable cities is ready since 1988. It is an application of catastrophe theory, graph theory and semiotics. It coincides with the ascension of OO programming and yet I cannot implement it as I wanted because I need prototype and aspect based languages like the subjective version of Self, called Us (Smith and Ungar, 1996)! The project was cancelled in the name of Java in 1995!

Couldn't they live together side by side at least?

Now I will cite Martin Fowler, another agile software developer: *Most software development is a chaotic activity, often characterized by the phrase "code and fix". The software is written without much of an underlying plan, and the design of the system is cobbled together form many short- term decisions. This actually works pretty well as the system is small, but as the system grows it becomes increasingly difficult to add new features to the system. Furthermore bugs become increasingly prevalent and increasingly difficult to fix. A typical sign of such a system is a long test phase after the system is "feature complete". Such a long test phase plays havoc with schedules as testing and debugging is impossible to schedule.*

Hence we need a breakthrough in software. A new consciousness that enable us to build alive systems that evolve gracefully independent of the novelty ahead simply because one knows how to deal with it! This is exactly what the semiotic square of Greimas teaches us to do. It is a representation of the elementary structure of signification (as defined in 2. Nature and culture) in the form of a double relation of disjunction and conjunction. (Figure 2 ).

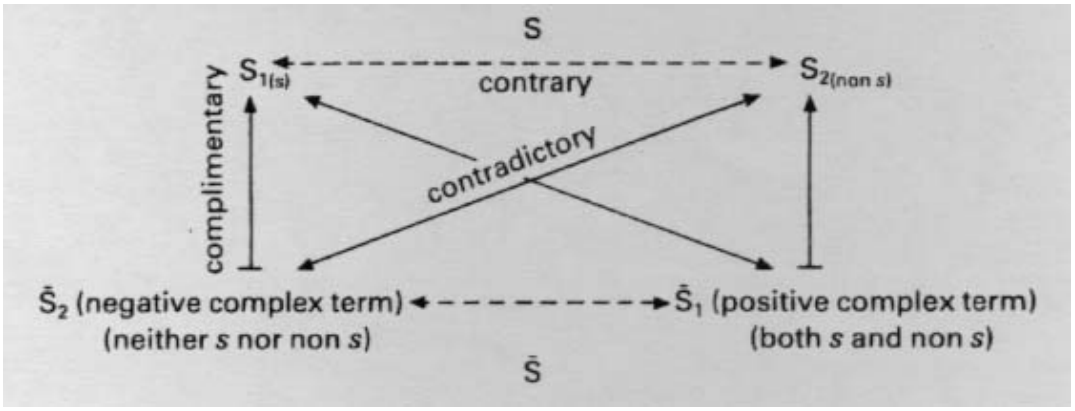



Figure 2.Greimas's semiotic square. Elementary structures of meaning can be formulated as semic categories, being able to be articulated as the semiotic square. These belong to the deep level of the fundamental semantics in the generative trajectory of the discourse. The structure of a tale also resembles the semiotic square.

What doubles the relationship are the complex terms on axis S: the positive and negative complex terms (Greimas also describes them as the complex term and the neutral term).There are two aspects of the semiotic square that constitute its importance. The first is that it exhausts, logically the possibilities of opposition in a schema which maps out the combinational relationships of those possible oppositions. Like language, it joins contrast and combination. The semiotic square describes what Greimas takes to be the logical structure of reality itself, that it presents fundamental categories of that reality. Its second feature is that it simultaneously inscribes, within this logic, a semantic component which, as we have seen, Greimas distinguishes from the 'pure syntax' of logical or mathematical constructions. We could say it contaminates the purity of logical syntax with its zones of entanglement that encompass the possibility of change, of content, within its structure. In sum, Nancy Armstrong narrates the generation of the square: *Once any unit of meaning [$S_1$] is conceived, we automatically conceive of the absence of that meaning[$S_1$], as well as an opposing system of meaning [$S_2$] that correspondingly implies its own absence [$S_2$].* In Figure 2, S represents the minimal unit of semanticism, the seme.

In terms of colours, black and white are minimum signifying units; thus they are simple semantic investments which as Greimas says in a different context in Structural Semantics, *constitute privileged cases…too close, if we may say so, to the structures of signification.* The center of his Semantics, Greimas argues, resides in the naïve hypothesis that starting from the minimal unit of signification, we can succeed in describing and organizing continually broader, larger ensembles of signification. This minimal unit, however, which we have called seme, has no existence on its own and can be imagined and described only in relation to something that it is not, inasmuch as it is only part of a structure of signification. Black is a semanticism and a seme. (Semanticisms usually are bundles of semes). Figure 3 can be inscribed with colours, apprehended as a relationship and presented as a semantic axis.

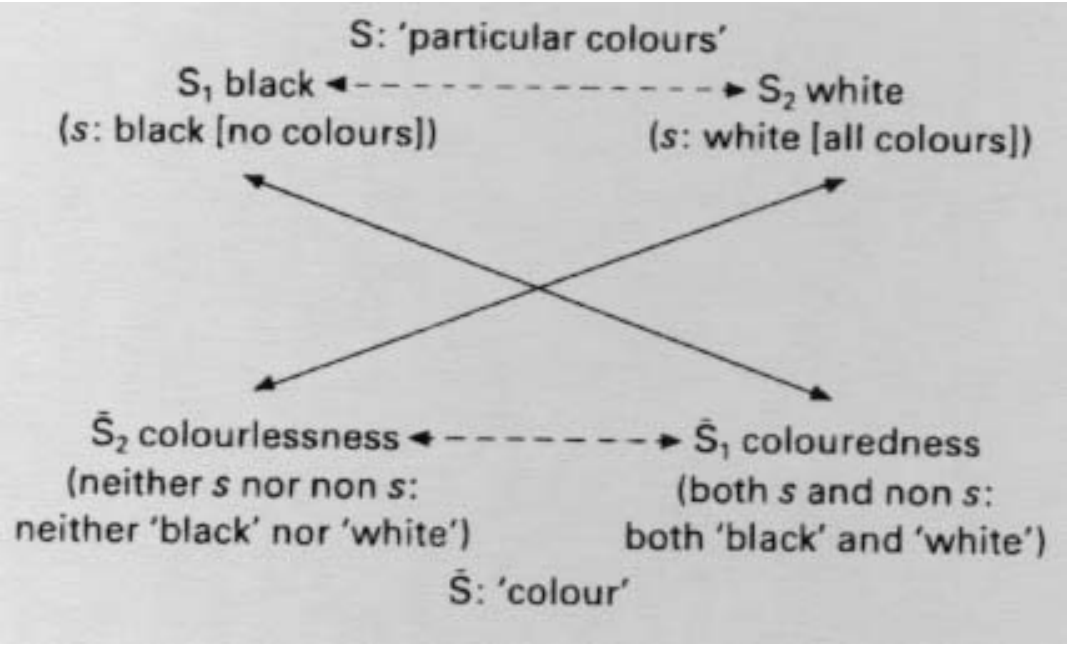



Figure 3. The semiotic square begins the generative trajectory of discourse and also introjects into the final text. For example Aspect oriented software development can be thought as separation and composition of concerns as the basic elementary structure, then articulated as parallelism and crosscutting of concerns.

The automatic conception of the absence of meaning is an implication: thus the levels of the square S vs S like the levels of language (e.g. distinctive features vs phonemes) are not in a relationship of reciprocal presupposition, but in a relationship of implication or direct presupposition (coloredness presupposes particular colors).

Semes combine to form lexemes, minimal functional signifying units (most often words, morphemes, but also inflections, suffixes, etc called the units of the first articulation) exactly like the phemes form the phonemes, minimal functional (i.e. realized) sound units. Black and white are lexemes that approach the status of single semes. A word like girl is a bundle of semes: /human/, /feminity/, /young/ etc.Greimas analyses *high vs low* and inscribes them in the semic system of spatiality. He describes the relationship existing between the semic system and the lexematic manifestation (Figure 4)

| Semes / Lexemes | spatiality | dimension-ality | verti-cality | horizon-tality | perspec-tivity | laterality |
|---|---|---|---|---|---|---|
| *high* | + | + | + | − | − | − |
| *low* | + | + | + | − | − | − |
| *long* | + | + | − | + | + | − |
| *short* | + | + | − | + | + | − |
| *wide* | + | + | − | + | − | + |
| *narrow* | + | + | − | + | − | + |
| *vast* | + | − | | | | |
| *dense* | + | − | | | | |

Figure 4. The Multi-dimensional separation of concerns should mirror this reasoning. It is very easy to unravel this way the semic categories.

Hjelmslev's articulation principle of the two plans evidences the function of the sign with two planes, the plane of the form (Hjelmslev, 1966)
Greimas delves deeper into the content plane of language, the realm of the signified. Having applied methods of linguistics to semantics enabled him to develop a sense of the palpable surfaces of things and the play of the surfaces. Hjelmslev views the substance of both planes as physical entities (sounds in the expression plane, things in the content plane). Greimas argues that the substance of the content is not an extralinguistic reality – psychic or physical – but a linguistic manifestation of the content, situated at another level than the form.
Quoting Hjelmslev: *The meaning which each minimal entity [morpheme] can be said to bear must be understood as being purely contextual meaning. None of the minimal entities, nor the roots, have such an independent existence that they can be assigned a lexical meaning..there exist no other perceivable meanings then contextual meanings; any entity, and thus also any sign [lexeme] is defined relatively, not absolutely, and only by its place in the context. From this point of view it is meaningless to distinguish between meanings that appear only in the context and meanings that might be assumed to have an independent existence (1961:44-45).*
Hjelmslev inspired me to model the architectural design as a language consisting of two planes: function and form and the stratas substance of the form and form of the form (Lourenci, 1988).
The software developer must be thinking how can I introduce this into modeling the real world? No problem at all, figure 5 shows how I applied this to model the sustainable architectonic object.

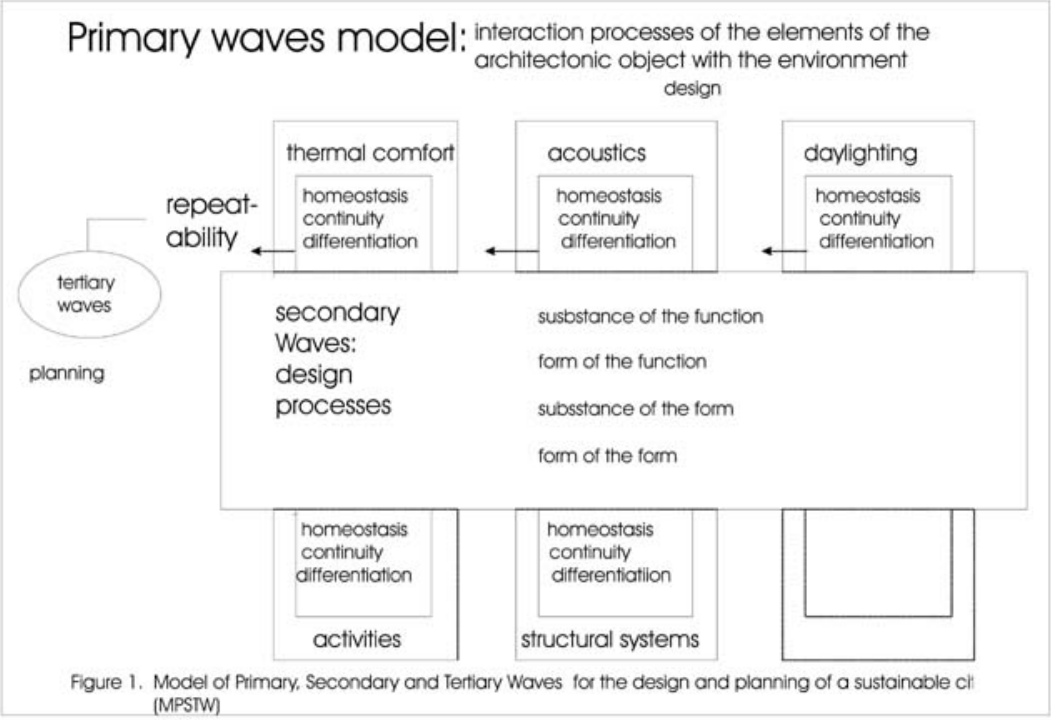


Figure 1. Model of Primary, Secondary and Tertiary Waves for the design and planning of a sustainable ci (MPSTW)

Figure 5. The nuclear semes are thermal comfort, acoustics, daylighting, activities, structural systems, building systems, beauty, etc. The contextual semes dependent on the context design are those connected to the design processes and those dependent on the interaction of the architectonic object with the environment are homeostasis, continuity, differentiation, repeatability. There is no isolated seme. Neither a collection of semes. There is an arrangement of semes ranging from the possible different manifestations of the elementary structure of meaning to the more complex structural groupings, connecting among them semes that belong to relatively independent systems. This kind of separation of concerns naturally triggers the emergence of crosscutting concerns. The latter little by little weave the composition of concerns. The parallelism of concerns is clearly gathered in primary, secondary, tertiary waves. Notice the introduction of the semiotic square structuring the general plot of the ecodesign model.

Obviously the opposite of sustainable architectonic object is a building conceived within the paradigm of the Modern Architecture movement. It is important to think in opposites to make explicit the semic categories as shown in Figure 4.

While the modern architectonic object lies isolated in an heterotopic environment, the sustainable architectonic object defines the urban ecosystem and is defined by it! Likewise any green product must be modeled to be conform to the new conceptions introduced by the Brundtland Report in 1987 by UNO known as sustainable development.

The search for the semic categories such as thermal comfort, acoustics, daylighting, structural systems, activities, etc. Obviously here we have multi-dimensional separation of concerns already! And the possibility of adding more semes. These are called nuclear semes and are equivalent to the core of the lexical meaning , independently of the context. However these semes are also lexemes (words) and can be analysed further in their relationship to two contexts, design and environment. These are called contextual semes. Namely, design semes, substance of the function, form of the function, substance of the form, form of the form, environment semes: homeostasis, continuity, differentiation, repeatability. Homeostasis in structural systems means basically the building will lie on a terrain and to keep its homeostasis, it is necessary a geo-engineering mapping to be sure the best terrain able to support the foundations with minimal cost has been chosen. Continuity is geological study and so forth.

Well but how is this generative? Let us examine this! But before I need to sharpen terminology examining Greimas's own words about nuclear semes and contextual semes.

Greimas works from the inventory of meanings supplied by the dictionary and organizes them to reduce and structure the occurrences of head into invariant and variant elements. Invariant semes like extremity and superativity for the lexeme head are the nuclear semes. Verticality and horizontality in the occurrences *to be in over one's head* and *head of line* respectively, generated by the context are the classemes, the minimal units of the semantic level found across at least two lexemes. A realized meaning-effect called sememe is the combination of nuclear semes and classemes.A sememe is the juncture of the semiological and the semantic levels of language, a double articulation works as a lexeme considered only on the plane of the content.
Hence the set of semic categories subdivides into nuclear semes and classemes. Any manifestation unity consists of at least two semes.
The procedure to analyse a lexeme from the viewpoint of its semes consists simply in extracting a nuclear seme in the first phase from a lexeme and then a classeme or context class that matches the lexeme.
Lexeme = nuclear seme + classeme, rather a sememe.
Now a puzzling example! Let's examine the dog barks. The contextual analysis of bark to detect the nuclear seme does not add more information due to the previous example with head, but its context refers to a cry that reveals the existence of two contextual subject classes that may match with bark. The animal class and human being class. Two contexts emerge depending either on the animal seme or the human being seme such as animal cry or human cry as in the boy barks to the moon!
Hence the grammatical structure is composed by semic categories that are not at all original and are realized in all sorts of sememes.
Manifestation can be defined as a combinatory of sememes. To define sememe as a manifestation unity, it is the same as introducing a new syntactic combinatory whose unities are the combinable elements. The provisional syntagmatic unit is a segmentation (actant) that combine in discourse.
I will not delve deeper because I suspect the reader has already internalized this generative approach. Or a way of deriving infinite combinations from finite elements.
Composition of concerns is achieved in architectural design obviously through geometric modeling. Three-dimensionality emerges out of crosscutting concerns. Here the geometric modeling works as a weaver.
Figure 6 shows the separation of concerns as well as the waves or levels of the Model of Primary, Secondary and Tertiary Waves to design and plan sustainable cities, generated from the scratch by the semes or its elements thermal comfort, structural systems, etc. The seme repeatability is responsible for the introjection of the amount of energy and materials spent in a single architectonic element into planning concerns.

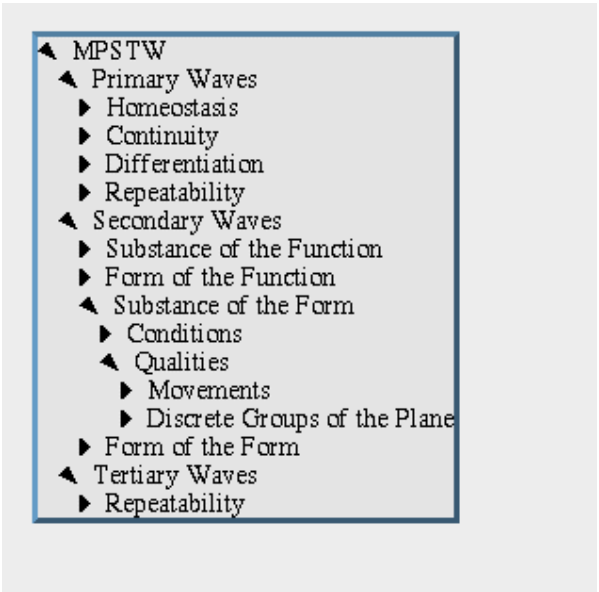
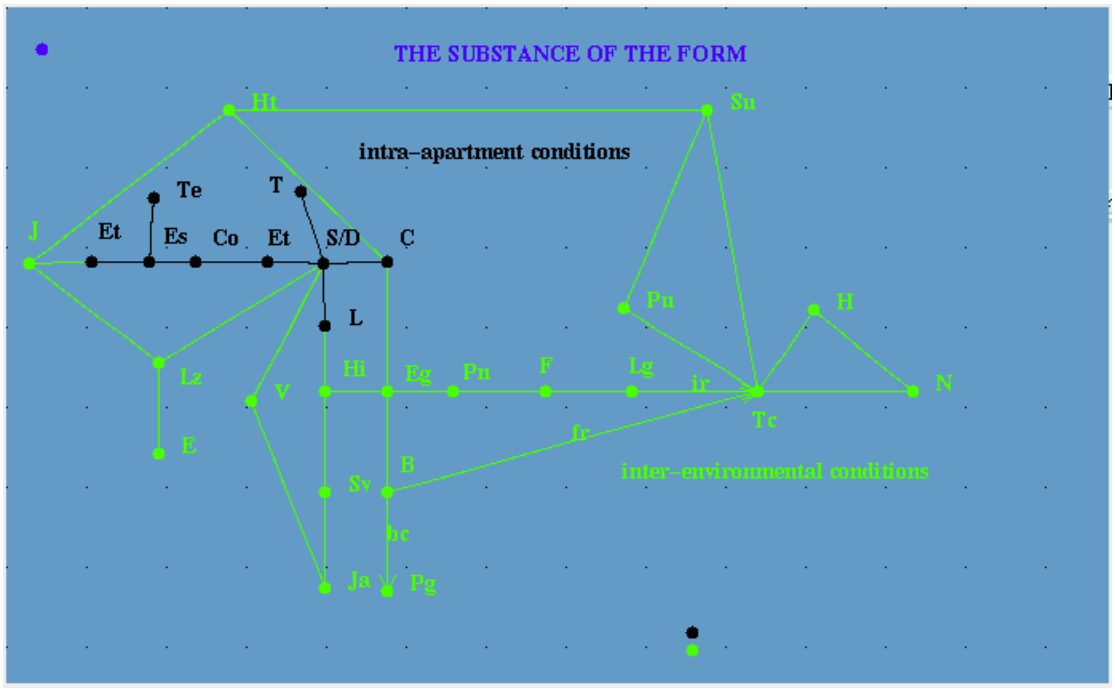



Figure 6. The whole basis of the architectonic object is applied recursively again and again to generate autopoietically the neighborhood unit, the borough, the city, etc.

Figure 7 gives a glimpse of how it is tightly coupled to the generation of the urban ecosystem.




B – biodigestor bc – biofuel C – eating area Co – corridor D – adult sleeping area E – sport area Eg – sewage Es – ladder Et – entrance area F – sand filter fr – fertilizer H – local agricultural area ir – irrigation J – garden Ja – greenery in wall L – lavabo Hi – Hygieniezation Lg – lake Lz – leisure area N – nature o/m – organization and maintaining area P – circulation area Pg – gasoline station Pn – natural or built wetland Pu – cattle S – social area Se- living room Su – supermarket Sv – laundry T- working area Tc – agricultural land Te – terrace V – varanda

Figure 7. This concern stresses the pluridimensional and topological nature of the architectural design. Its network of holistic relations. Or how the urban ecosystem is defined by ecology of the behaviour of the human being along the life cycle corresponding here to the structures that shelter this behaviour and how this defines the architectonic object and its interaction with the environment (Lourenci, 2001).

Figure 8 shows the elements of the graphical editor implemented in the prototype based programming language Self by Jecel de Assumpção and me within the context of X-programming. It enables us to mimic free-hand sketch which is 90% of the activity of architectural design.

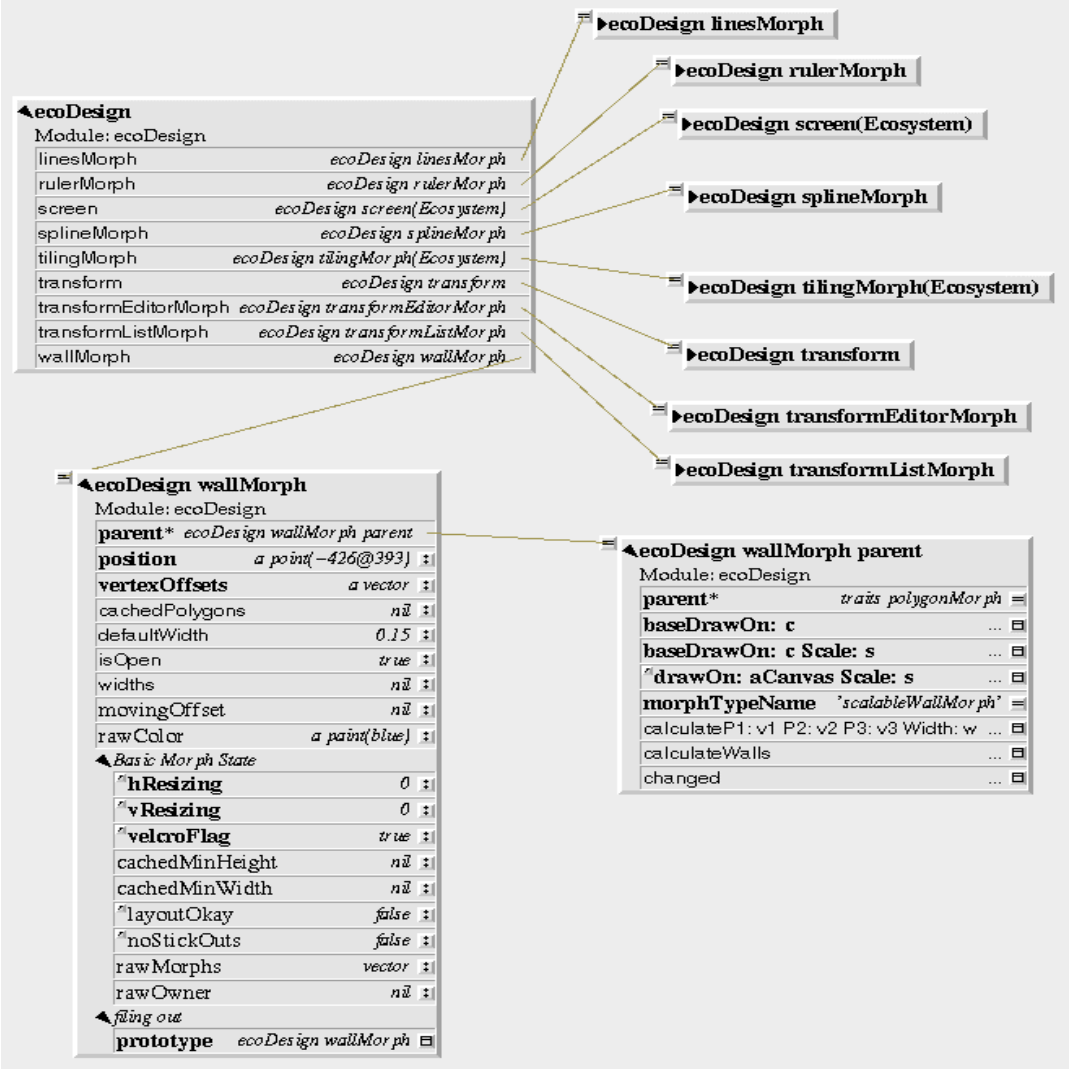



Figure 9 shows how we shape a free plan inspired by Escher's prototiles (figure 10) (Coxeter et al, 1988) that are of different shapes, differently from those traditional ones from the theory of tilings in Mathematics.

We first design the furniture for an apartment of single, middle class, then we start studying the ecology of its behaviour inside the apartment, shaping settings for sleeping, eating, washing, studying and so on. Finally we figure out the necessary shape for one apartment floor plan.

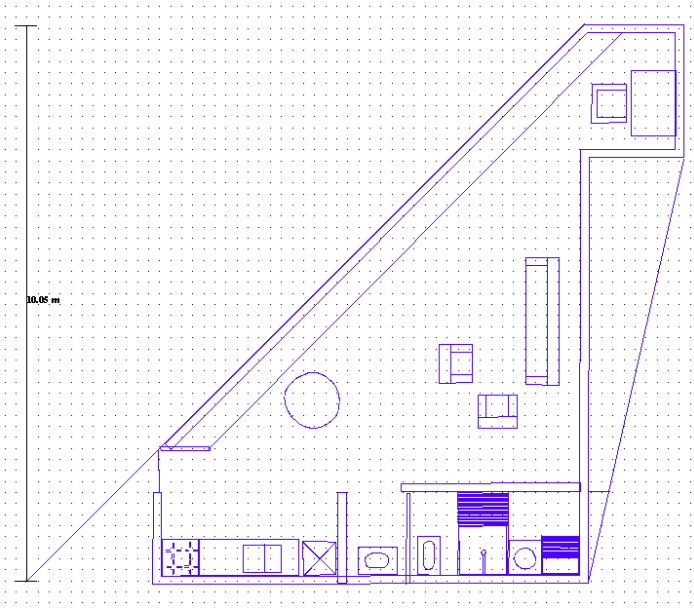



Figure 9. A free plan is generating by laying out the furniture mimicking the ecology of the human behaviour in the daily activities. Then you "dress" the lay-out with an unexpected shape!

Figure 10. M. C. Escher's tilings. A tiling is a set of figures that fill the plane without overlapping or leaving voids between them. a) . crystallographic group p4, generated by two rotations of 90 degrees, where the fundamental region consists of two starfishes, two shells, two green snails and a brown snail.
b) Crystallographic group p3m1, generated by three reflections in the sides of an equilateral triangle where the fundamental region consists of half a bird, half a fish and half a lizard. 

Next we choose p4m as the crystallographic group to shape the floor plan of a story of a building (Figure 11).
This apartment has a varanda of eight meters long and shelves up to thirteen meters in the living room. It is planned for someone who wants to integrate the activities totally. The alternative options are isolated activities (such as two-story apartments) or semi-integrated activities.
Of course the person may not appreciate this shape. The subgroup relationships of the crystallographic groups not only allows us to satisfy this user but also is responsible for the vertical integration or composition of concerns. It is currently under study,

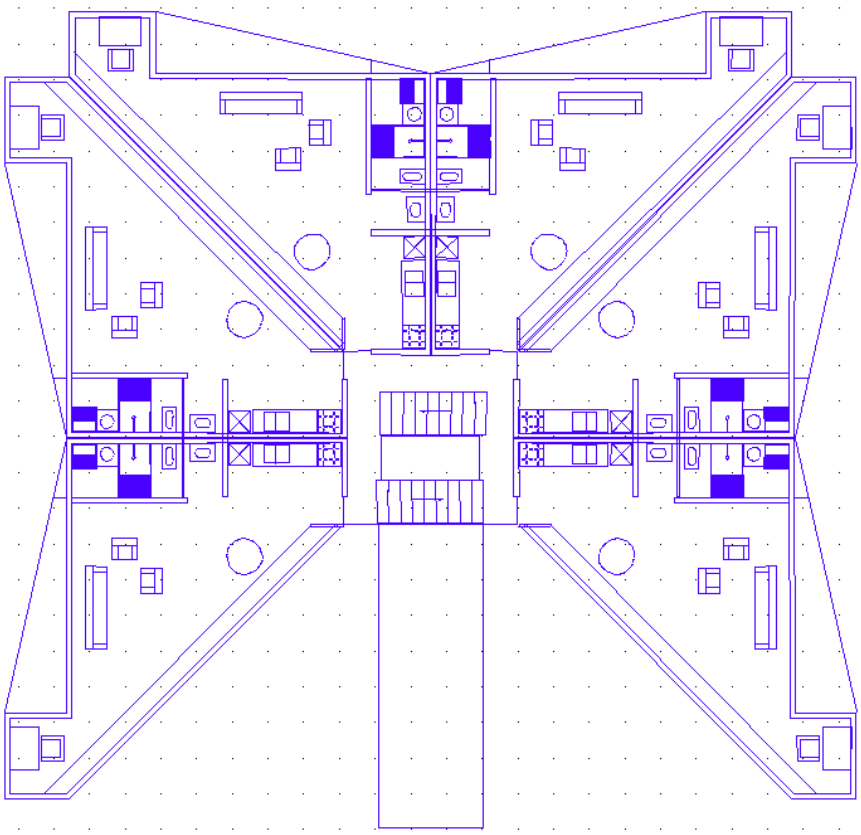


Figure 11. The crystallographic group p4m is chosen to give the final shape to the plan of apartments of a typical storey of a building.

**6. Conclusions**

The language game works in symbiosis with the human brain. Considerations about cognitive processes claims expression is not compositional formal encoding that mirrors a compositional conceptual construction. . This kind of process begins with two input spaces and produces a third space in which cognitive and linguistic work is accomplished. This third space –the blend- is both less and more than the two spaces. It is less in taking only partial structures from each of the two input spaces. The third space is also much more than the two input spaces: it has information about the two contexts and a frame for the current event or process being taken into consideration that is absent from both input spaces. The fourth-space of the many-sided space model is the generic middle space, a skeletal space that contains structure that is taken to apply to both of the input spaces. The construction of blended spaces is involved in reasoning, imagination, action, emotion and expression. Blending is a general cognitive operation, operating over categorization, the making of hypotheses, inference and the origin and combining of grammatical constructions. As example this explains why the desktop metaphor works.

Greimas seems aware of this.No one he includes in his semiotic project the reasoning and the feeling man. Likewise the Agile Software Development. Hence the inclusion of people as first order software components is a must (Cockburn, 2001).

This leads to a differentiation in the different levels of the alive software object similar to the generative trajectory of discourse. I detailed this better in (Lourenci, 2002b) because it demands the demonstration of how one builds a modeling that mirrors real world. This new vision triggers new ways to conceive software architecture and programming languages necessarily.

**Acknowledgments:** Special thanks go to Solveig Bjornestad, Alistair Cockburn, Dave Ungar, Harold Ossher, Jecel de Assumpção, João Antonio Zuffo, Jim Coplien, Juan Ramil, Manny Lehman ,Randy Smith, Vera M. F. de Lima, Winfried Nöth for the interaction and prompt disposal of necessary epapers. First CNPq, then FAPESP has been funding this research.

**7 References**


Assumpção, Jecel M. de: http://groups.yahoo.com/group/self-interest/links

Assumpção, Jecel M. de: Self/R. http://www.merlintec.com:8080/software

Cockburn, A.: Characterizing people as non-linear, first-order components in software development. http://www.CrystalMethodologies.org/ 2001

Cockburn, A.: Agile software development. Addison Wesley 2002.http://members.aol.com/humansandt/crystal/game/getasddraft.htm

Cooper, G.M., 1997, The cell. A molecular approach. Oxford University Press

Coplien, J.O. and Zao, L: Symmetry and symmetry breaking in software patterns. Proceedings of the Second International Workshop on Generative Components Programming 2000

Coplien, J.O.: It's time to kill software engineering. Keynote at XV SBES Simpósio Brasileiro de Engenharia de Software –3/5 outubro, 2001 Rio de Janeiro Brazil.

Coplien, J.O.: On the Nature of the Nature of Order. The Chicago Patterns Group presents J. O. Coplien on Christopher's Alexander latest work entitled: The Nature of Order In http://www.enteract.com/~bradapp/docs/NoNoO.html or http://www.rcnchicago.com/~jcoplien/bibliography.html

Coxeter, H.S.M. et al (eds): Escher: Art and Science. Proceedings of the International Congress on M.C. Escher. Rome, Italy 26-28 March, 1985. Eds. Coxeter, H.S. M. et al. North Holland 1988

Dear, M.J., 2000, The postmodern urban condition. Blockwell Publishers.

Emmeche, C. and Hoffmeyer, J., 1991. From language to nature- the semiotic metaphor in biology. Semiotica 84 (1/2): 1-42 http://www.nbi.dk/~emmeche/cePubl/91a.frolan.html

Fowler, M., 2001, The new methodology. http://www.martinfowler.com/articles/newMethodology.html

Gabriel, R., 2001, FeyerAbend Workshop. EuroPLoP2001 http://www.dreamsongs.com/

Greimas, A.J. and Courtés, J., 1982. Semiotics and Language: An analytica dictionaryIndiana University Press Bloomington 1982. Translation of Sémiotique:Dictionnaire raisonné de la théorie du langage, Librairie Hachette, Paris 1979

Greimas, A.J. et Court, J. 1979 Sémiotique. Dictionnaire raisonné de la theorie du langage. Paris

Greimas, A.J., 1966, Sémantique structurale. Larousse.

Greimas, A.J.1991. Sémiotique des passions. Des états de choses aux états d'âme. Seuil

Heidegger, M. 1952, Deer Ursprung des Kunstwerkes in Holzwege. Vittorio Klostermann

Jakobson, R. (undated). Life and language. The Hague:Mouton [Published in an issue of Linguistics: An International Review] or 1973. Main Trends in the Science of Language. London George Allen & Unwin Ltd

Joergensen, S.E., 1992, Integration of ecosystem theories: a pattern. Kluwer Academica Publishers

Lehman, M.M. and Ramil, J. F.: Software Evolution. Proceedings from the IV International Workshop on Principles of Software Evolution. IWPSE 2001, Vienna, Austria September 10-11, 2001

Lourenci, A., 2002. An evolutive architecture reasons as a semiotic, hermeneutic and autopoietic entity. Proceedings from the IV International Workshop on Principles of Software Evolution. ACM to appear

Lourenci, A.: A generative architecture as an aspect, multi-agent prototype-based membrane. Submitted to the Journal of Software Systems. Special Issue on Software Architecture. Quality Attributes.

Lourenci, A.: A proposal of a prototype based object oriented knowledge system to design and plan sustainable cities. PHD Thesis in Architecture and Urbanism. Faculty of Architecture and Urbanism. USP. 1998

Lourenci, A.: New computational paradigms reflect the nature of an ecodesign model. The unfolding and modeling of artistic consciousness in the infoera. Postdoctorate Scientific Report I. FAPESP. October 2000. http://www.lsi.usp.br/~lourenci

Lourenci, A.: Spirit, Energy and Information: Essential Elements of the Urban Ecosystem. Master's dissertation. Department of Architecture and Planning. EESC – USP. 1988. Summary in English available.

Lourenci,A.2001 New computational paradigms reflect the nature of an ecodesign model. Scientific Report 1. FAPESP

Mason, C.E., 1964. Rainer Maria Rilke. Sein Leben un sein Werk. Vandenhoeck & Rupecht in Göttinge

Nöth, W., 2000b. Manual de Semiótica translated from Handbook of Semiotics (1990) EDDUSP

Nöth, W.: Semiotic Machines. Cybernetics and human knowing, o appear in 9 (2002)

Prigogine, I.and Stengers, I., 1984, Order out of chaos:man's new dialogue with nature. New York Bantan

Rice, B.: The Arrow system. http://tunes.org/ search for Arrow. 2001

Schleifer, R., 1987.A.J. Greimas and the nature of meaning.University of Nebraska Press

Sebeok, T. 1985. Peirce's concept of final causation. Translations of the Charles Saunders Peirce Society 17(4), 368-382

Sebeok, T. A.,ed. 1999: Semiotica. Volume 127 –1/4 Special issues in Biosemiotics.

Smith, R. and Ungar, D.: A simple and unifying approach to subjective objects in Theory and Practice of object systems Vol.2(3), 161-178, 1996 http:// www.sun.com/research/self

Turner, M. and Fauconnier, G. 1995 Conceptual integration and formal expression. Http://philosophy.uoregon.edu/metaphor/turner.html