A generative architecture as an aspect-multi-agent prototype-based membrane

Albertina Lourenci

Laboratory of Integrated Systems, Department of Electronic Systems Engineering, Polytechnic School, University of São Paulo, Avenida Luciano Gualberto, 158, Travessa 3 Butantã, São Paulo SP BRAZIL, CEP: 05508-010

Abstract

Agile software development as well as other recent trends stresses the need for a radical rethinking of the software object likening its nature to the semiotic object. The Multi-Dimensional Separation of Concerns paradigm emerges as a result of the felt need for grasping the true nature of the software object. This brought a proliferation of approaches where the Babel effect is transparent. Hence I introduce Greimas's semiotic project, hoping to throw light on the true nature of software. The outcome highlights the need to focus on the various levels of the software system as differentiated to shape a unique language. And the architectural level emerges as a stage for narrative structures and discoursive structures, triggering the need to extend the prototype to cope with this demand. Both the generative nature of the ecodesign model to design and plan sustainable cities as well as its underlying associated cognitive processes brings to the fore the concern with functional quality attributes as well as non-functional blurring the difference between them. Its untamable nature dictates an efficient and highly expressive programming language like Self/US.

1.Introduction

Galal H. Galal and Tom Mens in a series of Workshops on Object-Oriented Architectural Evolution at ECOOPs together with the participants have been grasping that, on the one hand, in the conceptual view domain terms predominate as the main descriptors of the architecture, and it is where among others the impact of changes in requirements or the domain are studied

(Galal, 2000). On the other hand, changes to the software architecture can come from a variety of different sources: from changes in the problem domain, application domain, solution domain (emergence of new paradigms) and development domain (emergence of new paradigms such as agile software development). Moreover object-oriented software engineering principles such as design patterns, frameworks, delegation, aspects tailored to facilitate evolution tune in to ease transition from a software architecture to a software implementation. Hence architectural thinking should permeate the whole software design process, which is basically a learning process, where each new iteration leads to the refinement of architecture against a selected set of purposes or quality attributes. Non-functional requirements such as evolvability, reusability, scalability, flexibility drive the major part of software architecture. These are essentially related to cognition. The architectonic nature of the software object at each step must emerge clearly and shareable by all stakeholders when cognitive aspects such as semiotic, hermeneutic and autopoietic reasoning are observed (Galal and Tom, 2002).

Indeed Agile Software Development movement have been opening the gate to this vision brilliantly (Cockburn, 2002) insofar as eXtreme Programming (XP) can be likened to Jazz, where musicians create new tunes and idioms on the fly in response to fellow Jazz musicians. The issue of stakeholders comes up again here as the utility of architecture to various stakeholders needs to be evident, so that the stakeholders are motivated to participate fully in the architecting process.

Alistair Cockburn stresses "*a model as any communication is sufficient as soon as it permits the next person to move on with their work. Therefore, the work products of the team should be measured for sufficiency with respect to communicating with its target group"*. Moreover he also emphasizes that people resist change. They prefer to stick to old methodologies that don't work than to try innovation. However to face the challenges posed by problems mankind has been encountering even in the context of software development, tensions, setbacks and failures inherent in the development of any living being stimulate growth. Hence courage is central to

these methodologies. Socrates deems courage as the greatest virtue. Without it, all the others may become irrelevant in risky situations.

Another fundamental issue addressed by Kent Beck, Jim Coplien and I (Beck, 2000, Coplien, 2001, Lourenci, 2002) is that if software has been likened to art, the code is the material .*To respect the material means to hear what the material wants to be by itself, let him tell you what to do. Brancusi tunes in: it is while you sculpt…that you uncover the spirit of the material and its peculiar properties. Your HAND THINKS and follow the material's thinking.*This begs a question: what's the nature of the code? This led Coplien to the search for a new metaphor to replace software engineering: *the know-how of the "programmer" is concerned less with science than with craftsmanship and demands something like an art of writing.*

Curiously I am simply quoting Eric Landowski (Landowski,) if one replaces programmer for semiotician! He sees this decidedly as the methodological problem concerning semiotics. Consequently, I would state the right metaphor for the software object is the semiotic object! What's semiotics and how may it be related to software, if software is preferably art and music?

2. The software object and the semiotic object

First of all, semiotics like software is concerned with creating meaning understood by semiotic machines and does this as science and art. Winfried Nöth (Nöth, 2002) delves deeper into the notion of semiotic machines in his concerns with the computer not only as a machine restricted to the processing of symbols, but also involved in other sign processes. Hence machine semiosis: processes of sign production and interpretation within machines and between machines.

Peirce (1828-1914) defines semiotics as *the doctrine of the essential nature and fundamental varieties of possible semiosis, and semiosis as the intelligent or triadic action of a sign which involves a cooperation of three subjects, such as a sign, its object and its interpretant.* Peirce also assumes *the interpretant is….a sufficiently close analogue of consciousness.* Winfried Nöth describes the semiotic field from less to more complex semiotic sytems as a gradual

continuum from less complex to more complex processes of sign processing. Among the les complex processes are those merely mediated by instruments or technical devices such as a thermometer, a thermostat or the system of an automatic traffic light usually dealt with as S-types (specification types). The most complex processes of semiosis occur in living systems. Yes, there is semiosis in matter (crystals), machines, immunological systems and human minds. Moreover processes in which machines serves as mediators in human semiosis are certainly processes of genuine semiosis. Another genuine feature of semiosis that Peirce used to define is self-control. In systems theory the term autopoiesis is used to describe a system which evinces this kind of autonomy due to self-control. When the control comes from elsewhere, from outside, the system is an allopoietic system.

Today the distinction between allopoietic and autopoietic systems and more generally between engineering and biology is no longer as clear as it always seemed to be. On the one hand, doubts concerning the genuine autonomy of human consciousness have been raised. Free will is hard to achieve and one must be beyond genetic and cultural factors that determine behaviour. These evidences from evolutionary biology and contemporary genetics show the autonomy of human action and the destiny of humans are determined by factors independent of the self.

On the other hand, Nöth emphasizes that we are being confronted with the development of computer programs, automata and robots which no longer seem to be mere allopoietic artifacts but begin to evince features of autopoietic systems.

Although Peirce's semiotics opens the gate for reasoning about the semiotic object like like Eric Landowski does, his general theory of the sign (Lourenci, 2002) is still a semiotics of the sign! Hence unable to deal with semiotics as an art of writing and even as science and craftsmanship. Everybody would agree that a manual is not a literary work. What's the difference between Ulysses from James Joyce, which chapter on prostitution is inspired by his daughter's hallucinations and the latter? *Ulysses* likens to music! Peirce views a man like a living sign. Joys portrays the twenty-four hour reasoning of Ulysses! So we must create living signs in computer science! How? Main processes of holism and reductionism (analysis) have

been brought together! Well, the essence of evolution, since we are searching for natural evolving mechanisms to ease software evolution is generative reasoning!

Two constrasting trends have been becoming mainstream in OO community last decade. The design pattern movement spear-headed by Jim Coplien and inspired by Christopher Alexander's A pattern language and the Nature of Order (Coplien and Zhao, 2000) and novel approaches which try to find new dimensions for separation of concerns beyond the traditional concepts of module, class and object such as Adaptive Programming, aspects, composition filters, relations, (Bosch) hyperspaces, contracts, role modeling, subject-oriented programming, split objects , Us, a subjective version of the prototype based language Self, role modeling, activities and roles finally gathered under the umbrella of Aspect-oriented Programming (Bardou, 1998).

Curiously both target their methodologies at generation. Agile software methodologies come to the fore to contest the heavy software methodologies. The open issue is: Do design pattern movement and aspect-oriented programming fall where? It is up to the OO community to deem.

In the current well-known literature about generativity, there are the works *A Pattern Language and The Nature of Order* from Christopher Alexander in the realm of architecture, urban design and planning; *Sémiotique Dictionnaire Raisonné de la Théorie du Langage* from the semioticians Algirdas Julien Greimas and Joseph Courtés, *Catastrophe Theory* , René Thom, *The Geometry of fractals*, Mandelbrot, *Wholeness and the implicate order*, David Bohm, *Autopoietic Systems,*Maturana and Varela, *Introduction to non-equilibrium thermodynamics* from I. Prigogine, the xylographs from the Dutch artist M.C. Escher, especially concerned with tilings and the symmetry groups of the plane and the dotless plane to name a few.

The aspect- oriented prototype based knowledge system I am advancing to design and plan sustainable cities which when built function like living organisms depends on the progress of generative software systems that function like autopoietic machines (Lourenci, 1998).

Definitely from those *A Pattern Language* and *The Nature of Order* from Alexander and *Sémiotique Dictionnaire Raisonné de la Théorie du Langage* from Greimas are best suited to inspire us directly to generate software as a living organism. Curiously the former has inspired

the design patterns and the latter tunes with the gist of the search of the OO community for an integration of aspects, design patterns and components.

Both theories are concerned with wholeness, recognizing craftsmanship as a fundamental ingredient in model building. What exactly does this imply?

Alexander blurs the borders between architecture, urban design and planning brilliantly in a field where these areas are strongly compartmentalized leading to the current heterotopic environment of our cities and segregation in all dimensions.

Greimas contests strongly semiotics as a sign theory. He argues that *the still very vague, yet necessary concept of meaningful whole set forth by a message is crucial to semantics, and going beyond the narrow frame of the message. The conducting wire of the meaning (isotopy) is better conceived as the generative trajectory of discourse altogether - a trajectory which neutralizes the hierarchical opposition between immanence and manifestation. It fully takes into account the metalinguistic property of language, the fact that language uses any element and disrupts and neutralizes any hierarchy in its global aim of the articulation of meaning* (Schleifer, 1987).

He opens the gate to blur the difference between Eastern languages like Chinese and Western languages, or between prose and poetry or between verbal and non-verbal languages! Easing the understanding of the importance of the OO paradigm, which great merit consists in likening programming to how the world of the things that surround us is! The inheritance mechanism that more properly enable us to mimic the real world is delegation. More and more class-based approaches are trying to simulate the delegation mechanism, insofar as its sharing mechanism can be likened to aspect-weaving (Bardou,2000). A loop closes and yet there is no consensus about the relevance of sharing as delegation allows.

Why do I care? Because this mechanism simulates fundamental cognitive processes that enable us to simulate the most evolved semiosis, such as that intervening in the creation and interpretation of living signs (architectural design generated by symmetry groups of the plane, complex number, fractal dimensions, etc), the conceptualization phase of any design and free-hand sketch and so forth.

3. The Multi-dimensional separation of concerns (MDSCs) paradigm

The great criticism addressed by followers of the Multi-dimensional Separation of Concerns (MDSCs) paradigm has been identified as the tyranny of the dominant decompositions. Indeed like the class and even the prototype (object), tyrant decompositions such as agents (multi-agent approach), functions (in functional languages), rules (in rule-based systems), procedures (in procedural languages) allow the separation and encapsulation of only one kind of concern at a time (Ossher and Tarr,1999). MDSCs paradigm refers to the ability to identify, encapsulate, and manipulate only those parts of software that are relevant to a particular concept, goal or purpose. Many different kinds of concern may be relevant to different developers in different roles or at different stages of the software lifecycle.

In sum, the tyrant decompositions are a fake to grasp the network of holistic relations of the real world. Basically because the traditional education has not adapted to contemporary views like those portrayed by the literature cited above.

In OO programming a system is decomposed into several parts, independent of dealing with class or the prototype (object, despite all its flexibility); each of which is again decomposed into several parts , each of which is again decomposed until each part is simple and cohesive. The parts are implemented independently and composed to form the desired software system. Each unit of the decomposition is encapsulated in a class or in a prototype (Self). These units behave like pieces in a large puzzle. For an object-oriented system to fit into the category of MDSC, there must be classes or objects (Self) in different modules that represent separate aspects of the same entity. What are the advantages? Requirements-based modularization, decentralized development, unanticipated composition and software evolution.

Mehmet Aksit (Aksit, 1996) stresses that the proposed design patterns cannot solve these problems adequately because the composability features of the patterns are defined by the capabilities of the conventional OO model. Moreover he insists on that the so-called software-architecture definition languages (ADLs) try to model and structure higher-level design

concepts. But they do not adequately address the issue of evolution, separation and composition of concerns.

I feel here the need to sharpen terminology. Let us examine how Kiczales (Kiczales, 1997) tries to convey a neat vision of his proposals:

A component, if it can be cleanly encapsulated in a generalized procedure (i.e., object, method, procedure, API). By cleanly, he means well localized and easily accessed and composed as necessary. Components tend to be units of the system's functional decomposition such as bank accounts, and so on.

An aspect, if it can not be cleanly encapsulated in a generalized procedure. Aspects tend not to be units of the system's functional decomposition, but rather to be properties that affect the performance or semantics of the components in systemic ways. Examples of aspects include memory access patterns and synchronization of concurrent objects.

A GP-based implementation of an application consists of:

(i) a language,

(ii) a compiler (or interpreter) for that language,

(iii) a program written in the language that implements the application.

Likewise an AOP-based implementation of an application consists of:

(i.a) a component language with which to program the components

(i.b) one or more aspect languages with which to program the aspects

(II) an aspect weaver for the combined languages

(iii.a) a component program that implements the components using the component language, and

(iii.b) one or more aspect programs that implement the aspects using the aspect languages.

Just as with GP-based languages, AOP languages and weavers can be designed so that weaving work is delayed until runtime (RT weaving) or done at compile-time (CT weaving).[1]

What's wrong with this? A component means different things in different contexts, such as component based languages interested in entities larger than the class and the other related approaches to aspects do not use this designation for the same things. What the different methodologies try to model as aspects is even worse. They understand as properties the most different relations, structural, behaviour, domain-specific, quality attributes and so on! There are different definitions of aspects. Here it is implied especially cross-cutting concerns! But they can also be role models, activities, contexts, subjects, viewpoints, perspectives! Alessandro Garcia et al. (Garcia, 2001) refers to agency properties and agency aspects for agenthood! He defines agency properties are behavioral features that an agent can have to achieve its goals.And aspects should be used to implement the agency properties an agent incorporates such as interaction, adaptation, autonomy, affecting both core states and behaviors of agents. And also collaboration and learning aspects. Moreover Garcia et al extend the class concept to agents. They see objects and agents as complementary abstractions, additional features to objects transform them into agents.

Yet for those trained in semiotics and concerned with precision, which certainly is welcomed in the context of computer science this state of the art is at least confusing.

And since we are dealing with separation of concerns, of course the above presentation of Kiczales's approach sounds lucid! And tries to create an isomorphism between a

GP-based implementation of an application and an AOP-based implementation of an application .

The reader must remember it is exactly this sort of reasoning that we must be careful if we aim to simulate the real world! It is based on little knowledge of cognitive principles. We are not

[1] **Self, Smalltalk and LISP and CLOS are based on exploratory programming and hence at each step of the design the implemented code runs, so that when the implementation of the design is over, the implemented code makes reviews unnecessary (at each step if a problem happens, there is a debugger to check what is happening at the local of the bug) .**

stating this is bad. Absolutely, simply that this is not craftsmanship and may lead us far from our aims to liken software to be expressive as natural language, where more and more researchers are making clear that expression is not compositional formal encoding that mirrors a compositional conceptual construction!

4. The nature of the software object

To build an isomorphism between the different levels of the software object, such as the domain level, the architectural level and the code implementation level as suggested by analogical reasoning based on phenomenology seems not to be the best solution in the light of the above arguments. Indeed semioticians like Jean-Petitot Cocorda, who also is a mathematician, applying Catastrophe theory to his modeling of the speech tries to make clear how such a possible isomorphism would work. The main problem with this is lack of deeper knowledge of human cognitive processes as advanced by agile software developers. Conversely agile software methodologies have not delved deeper into the problem of treat phenomena as experience objects. This is the main reason I will try to describe it here briefly.Another reason is the need traceability of isomorphic reasoning structures from domain model to code implementation, which results perfectly modeled here. Most of the models having been developed so far lack this elementary concern.

What Alistair Cockburn has been courageously attacking are the heavy software methodologies: *One thing software development is not is model building. There has been a rash of people advocating model building over the last decade, to Ivar Jakobson's extreme statement: Software development is model building.*

Considering software development as model building leads us to inappropriate project decisions. If software development were model building, then a valid measure of the quality of the software or of the development process would be the quality of the models, their fidelity to the real-world, their completeness. However as dozens of successful project teams around the

world told me: "The interesting part of what we want to express doesn't get captured in those models. The interesting part is what we may say to each other while drawing on the board.

We don't have time to create fancy or complete models. Often, we don't have time to create models at all.(Cockburn, 2002:17-18).

This quoting evidences that indeed Jean-Petitot-Cocorda's modeling is not that bad, even if he doesn't include the human communication (Cocorda,)

While Kant insists on that while undetermined objects of an empirical intuition, the phenomena do not speak! They speak only when transformed through a semiotic and conceptual construction. Derived from theoretical imagination into objects.

A sense of object is determined by a network of relationships. . Here the relations are not categorical but holistic in the sense they are not independent, but are entities in a network always in expansion.

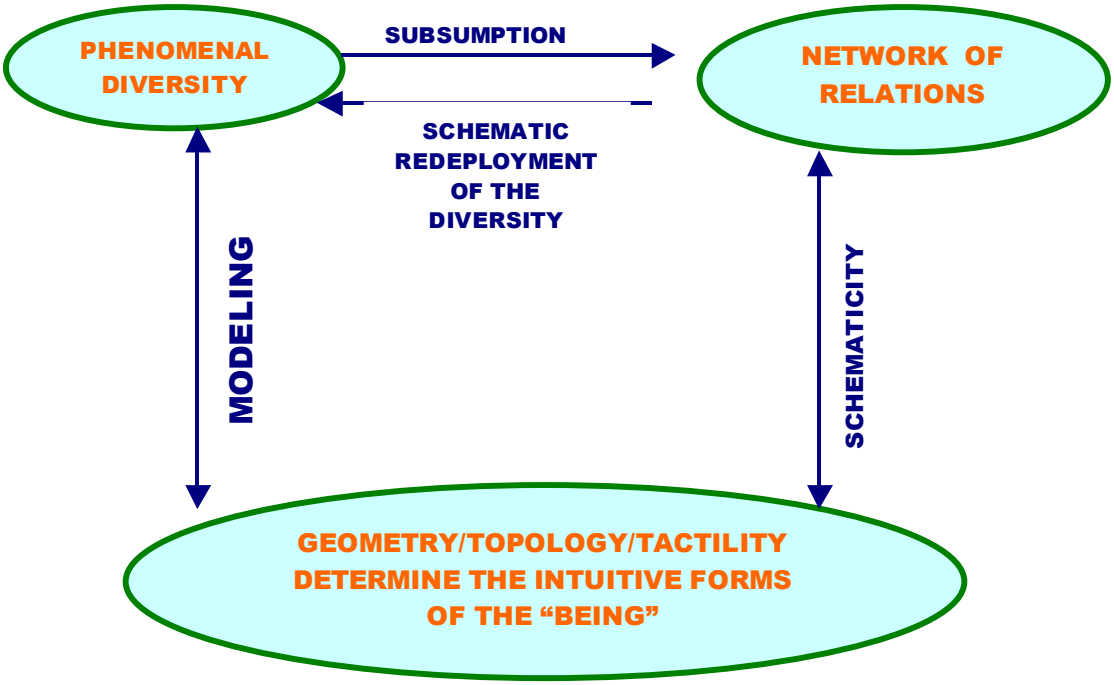
To subsume the phenomena of the considered region under this network of relations transform them into experience objects.

According to him Alexander's *A Pattern language* fails because paradoxically based on the principle that states the conformity to the things themselves, a model may not refer directly to the phenomena, but only indirectly (mediating it) through the network of relationships that subsume them (put then under a general principle). This does not erase Alexander's true contribution as put forward earlier! This factoring likens to its rational legitimation. But we have been insisting on analogical reasoning. Hence the models must be models that reflect the diversity of phenomena as stressed by Lehman and Ramil who proposes them as E-type (E for evolution) in contrast to the mainstream S-types (specification) (Lehman and Ramil, 2002).

However the subsumption under the network of relationships leads paradoxically from diversity to the unity of concept.

The models must deploy an internal diversity due to the meaning of the regional network of relations (Figure 1).

Moreover the words that describe this network of relations must be tailored in such a way to fit different contexts equally well. René Thom insists that here one must replace the meaning of the regional network of relationships by an explicit mathematical construction . Indeed the geometrisation of the concepts. One must spatialize the concepts insofar as to employ resources from geometric description – only this allows for true objectivation.



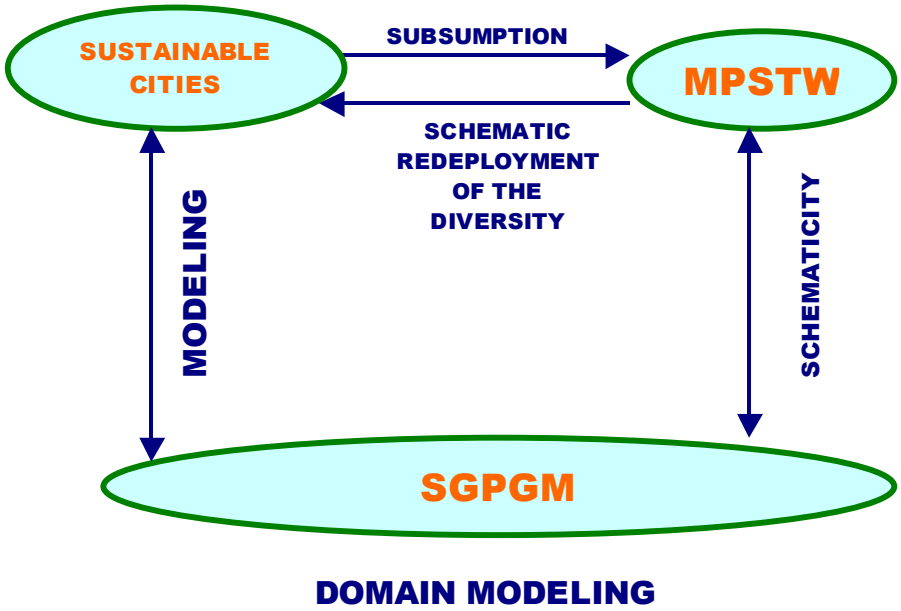


Figure 1. A knowledge system must be developed in a seamless process, trying to build isomorphic reasoning structures in all levels. However, visualizing the software object as a unique language adds to this structuring special considerations about conceptual blending and other cognitive processes such as collaboration and interaction and the nature of the natural language.

However one can transcend this geometric space and uncover diverse semantic spaces even more abstract where the concept can live! This geometrisation enable the reduction of the transpatial character of the concepts and the control of the analogy between model and reality. It is of the utmost importance for a model, not only its fitness to empirical reality but also its ontological bearing (its conformity to an objective essence).This step is known as schematicity. Kant defined it as construction of a concept in a mathematically determined intuition. Here one perceives the aim of the constitution of the morphological-structural objectivity. It changes the structural theories of theoretical types.

Hence *A Pattern Language* pastes too much to the empirical diversity without deploying a network of holistic relations and he only pursue his research uncovering geometric patterns in the Nature of Order (Coplien, 1997).

Only when we develop these contents both as a network of holistic relations and geometrically, one can model the physical reality with precision and perfection.

However Cocorda says nothing about modeling as a simultaneous and interactive game among things, human beings and language. To reach this in a model, one must portray the relationship between a subject and an object conceived as a narration that has to be implemented as software system. An aesthetic experience! Or exactly what architecture, urban design and planning needs to be to save Mother Earth!

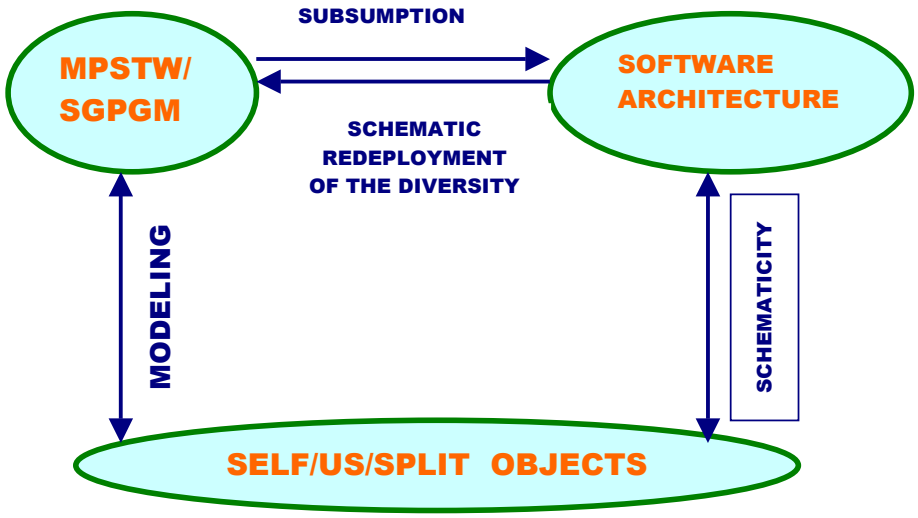
Jim Coplien highlights literature as the right metaphor for software. *It is how great programmers view it- mediocre programmers view it as engineering. Beauty matters. Richard Gabriel tunes in with his vision of the Erotic life of code.* (Coplien, 2001).

Up to date either we find the attitude of arguing laws from cases or of creating cases from laws: induction and deduction. And forget about the first phase of any scientific inquiry and interpretative strategy where abduction opens the gate to adopt an explanatory hypothesis. Curiously the definition of aesthetics is very near to abduction. The aesthetics which traces out the beauty is the sum of both, a search for laws and a search for cases. We design the beauty by

interaction with the object or by considering it a case from which we go towards a law, or thinking of laws and imagining cases which depend upon laws (Sung-do, 2001)

This is the guiding principles of my ecodesign model entitled The Model of Primary, Secondary and Tertiary Waves to generate sustainable cities **(MPSTW)**. Here the architectonic object defines the urban ecosystem and is defined by urban ecosystem. Not only the ecology of the behaviour of the human being along the life cycle, when h/she breaths, eats, defecates, studies, works, plays, prays, drives, travels, meditates and so on must be reproduced as first order functional quality attribute but also it must be mapped into space. Another first order functional quality attribute emerges: a geometry that not only reflects these diversified needs but also satisfy the expectations of a wide range of experts such as minimum use of energy and sustainable materials to guarantee Mother Earth works as Gaia (an ecosystem) due to the synergetic emergent arrangements that must also be beautiful to satisfy the requirement that architecture is above all art. Inspired by the tilings that M.C. Escher (Escher,) developed in his desire to express the infinite ranging from symmetry groups of the plane (crystallographic groups) and the dotless plane (similarity and conform groups, also fractals) I meet this goal through **the Symmetry Groups of the Plane Geometric Model (SGPGM) (**Figure 2).

Its generative nature is not being focused here, but rather that a similar reasoning like that from Kiczales about aspect oriented programming may be traced.



ARCHITECTURE MODELING

Figure 2. Although the **MPSTW/SGPGM** is a homo-iconic system (a system that reflects upon a homo-iconic system therefore reasons about structures of this single construct, all reflective discussions exist only in terms of collections of these fundamental objects in a homo-iconic system), and hence generative, other linguistic considerations are imposed upon it, having strong implications in the building of the architectural level.

I will reproduce here what Gregor Kiczales et al explained about aspect oriented programming and try to map isomorphically his reasoning structures to the domain model and its underlying geometric model.

A component, if it can be cleanly encapsulated in a generalized procedure (i.e., object, method, procedure, API). By cleanly, he means well localized and easily accessed and composed as necessary. Components tend to be units of the system's functional decomposition such as bank accounts, and so on.

An architectonic element such as activities, structures, environmental comfort, energy-generating structures, etc are the components of my ecodesign model.

An aspect, if it cannot be cleanly encapsulated in a generalized procedure. Aspects tend not to be units of the system's functional decomposition, but rather to be properties that affect the performance or semantics of the components in systemic ways.

All the interactions among the processes play the role of aspects which represent real cross-cutting concerns. The architectonic chords were likened to musical chords and are expressed through the geometric model.

A GP-based implementation of an application consists of:

(iv) a language,

(v) a compiler (or interpreter) for that language,

(vi) a program written in the language that implements the application.

A domain model of an application consists of:

(i) a domain model here the MPSTW

(ii) the designer's mind [2]

(iii) the geometric model orbiting around the symmetry groups of the plane and the dotless plane in this case

Likewise an AOP-based implementation of an application consists of:

(i.a) a component language with which to program the components

(i.b) one or more aspect languages with which to program the aspects

(II) an aspect weaver for the combined languages

(iii.a) a component program that implements the components using the component language, and

(iii.b) one or more aspect programs that implement the aspects using the aspect languages.

Likewise a symmetry-group-based implementation of the ecodesign model consists of:

(i.a) a multi-paradigm based ecodesign model with which to model the architectonic elements

(i.b) symmetry groups of the plane (rosette groups, frieze groups and crystallographic groups of the plane, similarity and conform groups of the dotless plane) with which to integrate or compose the architectonic elements

(II) a geometric weaver to compose the architectonic chords (role of harmony or vertical manifestations of music) through the subgroup relationships of the crystallographic groups and coordinate the dialogue of the free plan with the neighboring plans conforming to a smooth form metamorphosis

(iii.a) a Self program that implements the architectonic elements and

(iii.b) a multi-paradigm based program that implement the interactions, cooperative work and cross-cutting concerns [3]

[2] I highlighted the importance of a prototype-based programming language like Self because it allows a symbiosis between the computer and the human mind. This paradigm of interaction together with the cooperative work through the multi-user programmable virtual reality called Kansas in Self mimic the cognitive processes involved in architectural design perfectly well.

Apparently this arrangement is neat. But where is the place for architecture here? This begs a question. Where is the place for the human interaction considered as first order linear component in software (Cockburn, 2001). Well, where are the considerations about cognitive processes that as pointed forward before claims this isomorphism attempts should be regarded with distance: Expression is not compositional formal encoding that mirrors a compositional conceptual construction. I must remark here that Self graphical interface allows lots of respect for human cognitive processes such as interaction and collaboration due to Kansas, a multi-user programming virtual reality. It mimics perfectly well fundamental reasoning in design. It allows for cooperative work even through long distances.

5. Greimas's semiotic project: throwing light on meaning

Apparently most of the reasoning in OO community is based on conceptual metaphor: metaphor carries structure from one conceptual domain (a "source") to another ("target") directly. In this approach conceptual domain refers to a vast organization of knowledge, such as our knowledge or journey or dreaming. A conceptual domain has a basic structure of entities and relations at a high level of generality – for example the conceptual domain for journey has roles for traveler, path, origin and so on. A conceptual metaphor consists of a partial mapping of the basic structure of one conceptual domain (the source) onto another (the target). The two-domain model is actually part of a larger and more general model of conceptual projection. Mark Turner and Gilles Fauconnier (Turner and Fauconnier, 1995) call this new model the many-space model. The many-space model explains a range of phenomena invisible or untreatable under the two-domain model and reveals previously unrecognized aspects of even the most familiar basic metaphors. 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

[3] I hope the Self/R being developed by Jecel de Assumpção Mattos fulfills this need. See http://www.merlintec.com:8080/software, it integrates reflective abilities and aspects.

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 also 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.

The way Greimas unfolds his semiotics tunes with this conceptualization.

Of course it seems the abuse of metaphors has led to the so-called schizophrenia of the object and the known reaction as Multi-dimensional separation of concerns paradigm.

I delve deeper into Greimas's semiotics in this section because I feel it may help software developers to perceive why it is difficult to characterize what a substance or relation is.

Tracing a parallel it seems the OO community is at the same level as the semiotic community after the two wars. It was necessary to start the destruction of the sign. Now the destruction of the object and class begins.

Greimas modeled his minimal unit of signification the seme on the distinctive features of the phoneme of the Prague School, which invented and developed phonology between 1929 and 1939 as opposed to the pheme.

Every phoneme can be analysed as a collection of immanent features which are never realized independently but only in combinations within particular phonemes which create signifying differences in contrast with their binary opposites in the combination (bundle) of features of different phonemes. Thus the phoneme /t/ in English is a bundle of features (-vocalic, +consonantal, -grave, +diffuse, -strident, -nasal, -continuant, -voiced) which is identical to that of /d/ except the bundle of distinctive features in /d/ contains + voiced. All the phonemes of a language can thus be reduced to a combinatory of a much reduced number of distinctive

features. Such distinctive features – voiced vs unvoiced in the opposition of /d/ vs /t/ in English exist only in a structure (Schleifer, 1987).

In Structural Semantics (Greimas, 1966), he calls the distinctive units of signification semes. *The seme like the distinctive feature of phonemes, the pheme 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.* [4] (Schleifer, 1987:69).

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 3)

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

Figure 3. The Multi-dimensional separation of concerns should mirror this reasoning.

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).*

I hope the reader is perceiving the deeper Greimas's conception of the method of semantic analysis. In Hjelmslev's conception there can be no positive minimal units of signification because such units as aspects of the substance of the content are outside linguistic analysis. Thus the invariant elements of human perception – tactile, spatial, aspectual – are given *a priori* and not incorporated within the structure of language. It seems that a more global vision of the nature of the language is a difficult undertaking. Hence it is not a coincidence that what has been happening in semiotics is also happening in the OO paradigm.

Greimas adds to the Hjelmslev's articulation principle of the two plans, concerns about the deep levels: the unities with smaller syntagmatic realizations lie in a deeper language level, while the greater unities belong to the surface level.

A hypothesys of isomorphism may be presented under the form of a simplified schema:

Deep level: phemes

Expression plane

Surface level: phonemes -----syllables

Manifestation plan {realized phonemes, lexemes}

Surface level: sememes - semantic utterances

Content plane

Deep level semes

Apparently one may think of an isomorphism between the linguistic figures of the two autonomous planes , but indeed this isomorphism stops (except in the rare cases where the mono-sememic lexeme has just one phonemic realization) not only on the manifestation plane where to a realized phoneme correspond a lexeme, but also and above all in the level of the syntagmatic organization of language where the linear combination of phonemes results in syllables-unities, while a combination of sememes builds semantic utterances.

Obviously the isomorphism does not correspond to the homologation of terms to terms , phonic segment to semantic segment of the two planes.

The power of double articulation:

First articulation substance– manifested semantic universe

Content plane:

Second articulation: form immanent semantic universe: semiological level – semes in terms of invariants –nuclear semes

semantic level – variant semes- classemes

nuclear semes + classemes = sememes

Ronald Schleifer highlights that the combinations and contrasts of semes develop the network of holistic relations of signification and constitute what Greimas calls the immanent semantic universe. It is the second articulation of the content plane (just as phonology is the second articulation of the expression plane). The first articulation is the manifested semantic

universe.Within the immanent semantic universe Greimas articulates two levels of content analysis: the semiological level and the semantic level. The semiological level organizes the semes in terms of the invariants contained in particular lexemes while the semantic level organizes the variant semes called classemes.

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

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. The idea of unit is determined by the classeme discreteness: the manifested universe in its entirety Greimas writes constitutes a class definable by the category of totality. *This category which we propose to conceive, following Brondael as being articulated into discreteness vs. integrality divides the manifested universe by realizing, at the moment of the manifestation, one of its semic terms into two subclasses, constituted in the first case of discrete units, and in the second case of integrated units. Placing ourselves at the level of the manifestation of occurrences, we see that every sememe, overdetermined by the presence in its core by the classeme 'discreteness', is presented as a unitary object and produces, as its effect of meaning, the idea of substance – thing, person, image, symbol, and so forth. On the other hand, we see that every sememe having the classemes integrality presents itself as an integrated ensemble of semic determinations.*

The semantic universe, manifested as sememes, if considered as the class of the classes appears as an immanent syntactic universe, able to generate greater syntactic manifestation unities. Greimas designates the subclass of sememes defined as discrete units *actant* and the one that designates the sememes as integrated units, *predicate*.

The combination of a predicate and at least one actant is a bigger unit, which we call message.

The syntactic manifestation organized in messages appears as a new very simple combinatory. To complexify the syntactic manifestation, Greimas introduces the division of the class of the predicates, as dynamic and static ones. The static predicates inform about the states while the dynamic ones about the processes undergone by the actants. (Example: in the lexeme *aller, Cette robe lui va bien or Cet enfant va à l'école.)*

The dynamic predicates are called functions and the static ones, qualifications.

Thus the message while a combinatory of sememes, behaves either as a fonction or as a qualification on the one hand, and on the other hand as an actant.

All this entails the substantification of relationships. It is the fact that language apparently composed of radically relational elements creates the meaning-effect of discrete entities.

Nietzche asserts the importance to human life of attributing substantiality to phenomena: *the extreme case would be the man without any power to forget who is condemned to see becoming everywhere. Such a man no longer believes in himself or his own existence; he sees everything fly past in an eternal succession and loses himself in the stream of becoming.* This may explain the rejection to the prototype and the adherence to the class. This characterizes the phenomenon of experience. We experience a morphology of things and the paradox of the simultaneous contradictory fact that upon reflection these things disappear in the order of structured relationships. Hence paradoxically meaning can be both missed and apprehended, it can be figured out. Greimas attempts to account for both orders of facts, relational and substantial, the logic and morphology of discourse.

He situates these orders of facts on different levels, the semio-narrative level of discrete actants and the realized level of an apprehended meaningful whole.

Greimas defines linguistic activity in terms of messages and their algorithms. Yet a succession of messages he argues can be considered as an algorithm only if the functions manifested in it are all attributed to a single actant.

Nevertheless, there is an ambiguity here; if in individual messages predicates seem to be attributed to actants, at the level of discursive manifestation predicates are creators of actants, which are representative we should say of the classes of predicates. Hence its double status of actants as relational and substantifying, integral and discrete: as invested contents, the actants are, in fact instituted by predicates within each given microuniverse; as syntactic subclasses, they are however rightfully anterior to the predicates since discursive activity consists in the attribution of properties of these entities.

If actants can be conceived as both the result and the basis of predicate analysis (of the two kinds of predicates Greimas describes, functional predicates that describe activities and qualificative predicates that present qualities or states, then why does he choose as he does to

conceive of actants as discrete elementary units for analysis rather than integrated ensembles of other elements? Propp follows the latter course, making actants secondary to functions.

This is Greimas's semiotic project. The theoretical mediation between narrative forms and linguistic forms of sentential dimensions, what he calls an attempt to shed a little light upon the relations which can exist between discourse and the sentence, between discursive linguistics and sentential linguistics. For this reason, Greimas defines semiotics in terms of actants, actantial roles and the structure of the narrative functions of discourse in the terms of linguistic analysis. Actants are implicit abstract agents, a kind of grammar or structure of agency-effects in discourse analogous to the abstract (sentential) categories of syntax in the same way sentential categories (grammatical subject, object and so forth) are analogous to the combinatory of the implicit discrete distinctive features of phonology. Functions as classes of narrative action ae closer to the surface of discursive activity, less the abstract narrative form of actants than the raw material of narrative form.

As abstract agencies, actants are defined reciprocally in relation to one another in terms of their actantial roles and in relation to the narratives in which they appear in terms of their spheres of action or narrative functions.

Actantial analysis accounts for the given sense not only of meaningful wholes of discourse but also of the piecing together of meanings, the give experience of figuring out signification.

In sum, all these considerations enabled Greimas to propose the generative trajectory of discourse comprised of a deep level of virtual meanings present in disjoined or actualized elements on a manifest semio-narrative level before the realization of signification on the level of narrative discourse.

Thus he proposes an intermediary level between the possibilities or virtualities of immanence and the concrete realizations of apprehended meaning- a level of elemental but not global comprehension.

In these terms he conceives the dichotomy between immanence and manifestation as that between virtuality on the one hand and the double articulation of manifest signification *actualization and realisation* on the other.

Actualisation is the surface semio-narrative level of the actants and functions, mediating between deep level of immanent semantics and syntactics and apprehended discursive meanings in the same way that the semantic level and the realized meaning-effects of particular sememes. In the following diagram, Greimas visualizes the distribution of the diverse components and sub-components of the generative trajectory of discourse.

<table>
<tr><th colspan="4">GENERATIVE TRAJECTORY</th></tr>
<tr><td></td><td colspan="2">syntactic component</td><td>semantic component</td></tr>
<tr><td rowspan="2">Semiotic and narrative structures</td><td>deep level</td><td>FUNDAMENTAL SYNTAX</td><td>FUNDAMENTAL SEMANTICS</td></tr>
<tr><td>surface levels</td><td>SURFACE NARRATIVE SYNTAX</td><td>NARRATIVE SEMANTICS</td></tr>
<tr><td>Discoursive structures</td><td colspan="2">DISCOURSIVE SYNTAX
Discoursivisation
actorialisation
temporalisation
spatialisation</td><td>DISCOURSIVE SEMANTICS
Thematisation
Figurativisation</td></tr>
</table>

Figure 4. The generative trajectory of the discourse may be mirrored in my knowledge system to generate sustainable cities isomorphically, characterizing software as an autopoietic machine.

I hope this may shed light on the complexity of the language. If one does not proceed to deep analysis in relation to meaning, the outcome is a Babel effect, a proliferation of approaches as we have been witnessing where it is hard to grasp to what it is being referred.

6. Greimas's semiotics can be mirrored in the visualization of an aspect-multi-agent prototype based architecture

The first impact has already widened my horizons and I realize it makes sense to view the software object in all its level from domain model to code implementation as a single language.

Hence necessarily the architectural level plays the role of the stage where not only discourse through multi-agents may happen but also an intense narrative activity characterized by all sorts of aspects.

Exploiting Serres's work (Serres, 1986) where tactility becomes a model of cognition, we could view software architecture as a sensing skin of a body in the flow of a fluid, chaotic world.

This tactility is double: it is both a sensing and a sensed skin! It has a dialogical nature! It is its responsibility to create greater and greater meaningful wholes in regard to the domain model and the real world and the implementation code.

It has a reflective nature . The MPSTW/SPGM is a homo-iconic system. A system that reflects upon a homo-iconic system therefore reasons about structures of this single construct. All reflective discussions exist only in terms of collections of these fundamental objects in a homo-iconic system.

Contradictions between models in the system allow for the designers of the system to make assumptions contrary to the current state of information and study the consequences of the assumptions upon the relationships among various models.

Figure 5 shows the architectonic object and its nuclear semes and "classemes", responsible for the Multi-dimensional separation of concerns.

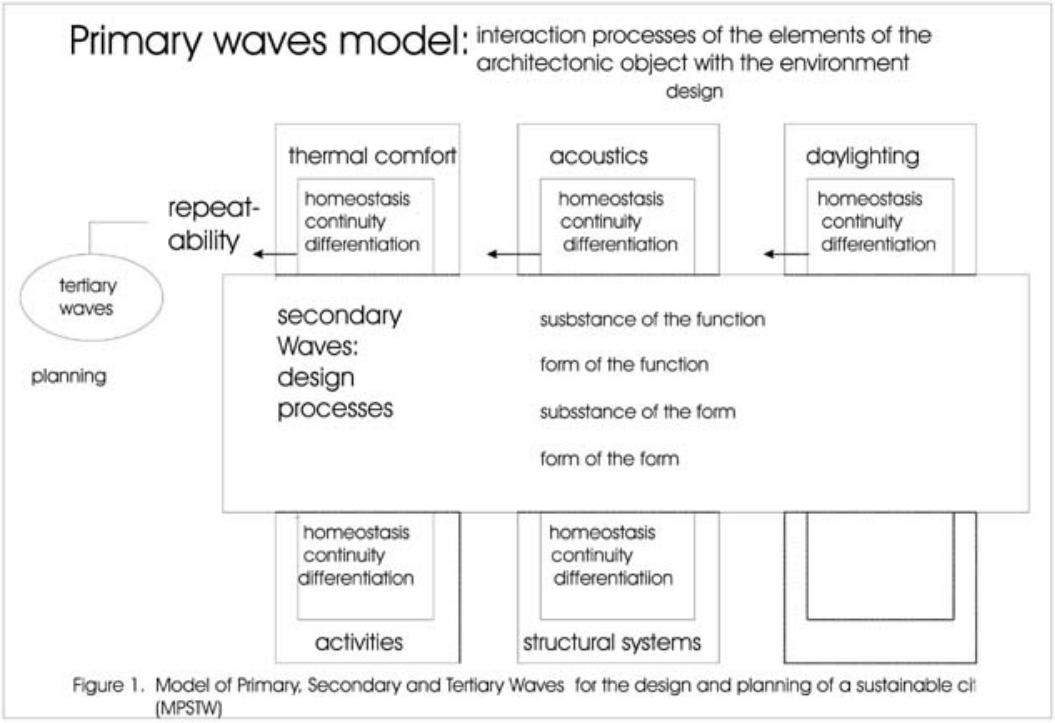


Figure 5. The nuclear semes are thermal comfort, acoustics, etc. Another class of nuclear semes is substance of the function, form of the function, substance of the form, form of the form.

The variant semes dependent on context are homeostasis, continuity, differentiation, repeatability. This leads to separation of concerns enabling sharing of work gracefully. Aspects characterize the interactions of these processes. It would be very important in terms of architectural design if the whole entity architectonic object could be implemented as a unit. Studies held by Bardou about Split objects shows this may be unreachable for a long time (Bardou,1998).

On the one hand, its regional categories subsume the phenomenal diversity involved in architecture, urban design and planning. However they are meaningless if not associated with the target phenomenal diversity. They describe separating all concerns involved in the act of design and planning neatly. There is a regional category for every dimension ranging from concerns with the ecology of the behaviour of the human being, the "thingness" of the

architectonic object revealed through its elements , the characterization of the environment (earth, climate, vegetation) to the topological/geometric processes involved in the act of design. On the other hand, a new geometric consciousness emerge from this substrate, mathematically translating into meaningful form (schematicity) the underlying phenomenal biodiversity of the target sustainable cities. The corresponding **Symmetry Group of the Plane Geometric Model (SGPGM)** composes each concern horizontally and vertically shaping a final integrated sustainable architectonic object (Figure 2).

My intention is always to trigger strong associations in the reader's mind. I hope the reader may visualize with great ease that the basic unit of my architectonic object depicted in figure 5 plays the same role as the semic category in the generative trajectory. It is the instance ab quo of the generative trajectory and belongs to the semantic component of the deep level. Its syntactic component lies at the level of the geometric model. I will not delve deeper into it here, but it has a generative structure based on the concept of prototile, tilings and crystallographic groups of the plane (Lourenci, 1998) (Figure 6).

In the paper entitled *An evolutive architecture reasons as a semiotic, hermeneutic and autopoietic entity* (Lourenci, 2002) I praise that this structuring leads to a model fond of change, evolution, cooperation, interaction, promoting infinite synergies. Any change or incremental need fits nicely into any level of the knowledge system not only due to its isomorphic nature pervading all levels but also due to autonomous organization of each level. I mean to boost a component of a level does not necessarily mean to propagate the change to all levels due to autonomy. However if it is decided the change will be stably integrated to the system, isomorphic structures pervading all levels ease this task superbly. Due to the possibility of independently experience with each regional category or process, one can design and implement it interactively, cooperatively or independently. If it passes the test, it may or may not be integrated to the whole model. Yet the model does not have the nature of interactive components or a list of elements because it has a geometric-topological nature and all components are linked to each other recursively or intertwined.

Hence obviously as I showed in the previous section how the semic category pervades the actantial structuration at the surface level characterized by the syntactic and the semantic components, likewise, the basic seed of the MOPSW and its underlying geometric modeling allows its expansion as a homo-iconic system. The substance of the form. (Figure 7) shows through its topological relationships how strongly the intra-apartment conditions are dependent on the inter-environment conditions.

This generative nature enables the development of a stable software architecture.

Or how "form follows function", I mean how function is transformed into meaningful forms such as eating area, sleeping area, leisure area, circulating area, hygienization area, etc inside the apartment world. It shows how the latter are thoroughly dependent on corresponding social and environmental structures.

The detailed sustainable sanitation systems shows clearly the connection of the apartment to the surroundings where rainwater can be stored in lakes or in the roof or tanks, infiltrated through swales with biologically active soil in the ground, recharging the groundwater or respecting the local water cycle (stormwater runoff is often loaded with a wide variety of organic and inorganic chemicals, hence direct infiltration into the soil should be avoided); separated

blackwater can be treated anaerobically in biogas plants combined with the digestion of organic household wastes results in a mixture that is suitable for this process; usage of an existing treatment plant without nutrient removal as irrigation water that carries also fertilizer demands integration of agriculture and landscape into cities; hygiiniezation of the effluent or crop restrictions may be necessary; treatment of wastewater or greywater from bathrooms, washing machines and kitchen (little nutrients) can be done by constructed wetland as wastewater lagoons or sandfilters with reed; the combination of treatment and agriculture can be applied with the system of energy forests; composting provides withlong-term fertilizer while biogas-systems or aerobic wastewater treatment produces fertilizer that should be applied during the growth periods only; etc.

For such a tight integration of sanitation systems of the apartment with the environment, one is already modeling the Tertiary Waves of the ecodesign model strongly. Hence evaluation of overall efficiency with tools such as LCA(lifecycle assessment) MIPS (material intensity per service unit) or SPI (Sustainability Index) may be unfolded in the Department of Energy under Prof. Lineu Belico's coordination.

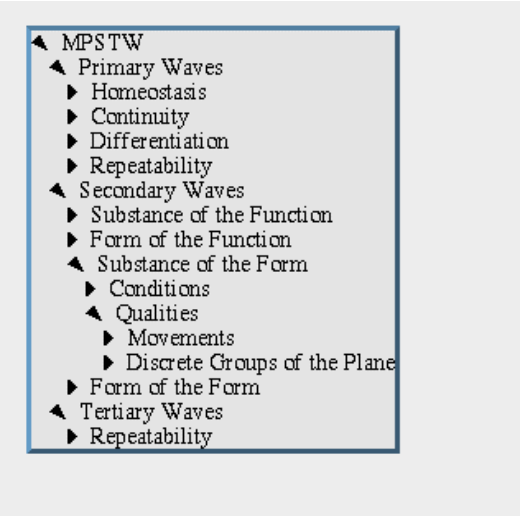


Figure 6. A new geometric consciousness emerge from this substrate, mathematically translating into meaningful form (schematicity) the underlying phenomenal biodiversity of the target sustainable cities.

A more encompassing notion to retrofit the wastes reaching the goal of zero emissions is the Integrated Biosystem. For a biologist, an integrated biosystem contains at least two biological activities or subsystems where nutrients in by-products (waste) from one sub-system serve as resources or inputs for another.

The integrated bio-systems approach follows three basic principles. The first principle is to use all biological organic materials and wastes instead of throwing it away.l The second principle is to obtain at least two products from a waste. The third principle is to close the loop for the material and nutrient flows to achieve total use of a resource and zero waste disposal. Its application ranges from situations where natural resources were limited and when the full use of resources is crucially interlinked with human survival, problems related to waste management and to improve industrial productivity to utilization and management of agro-industrial wates in industry (Foo, 2000).

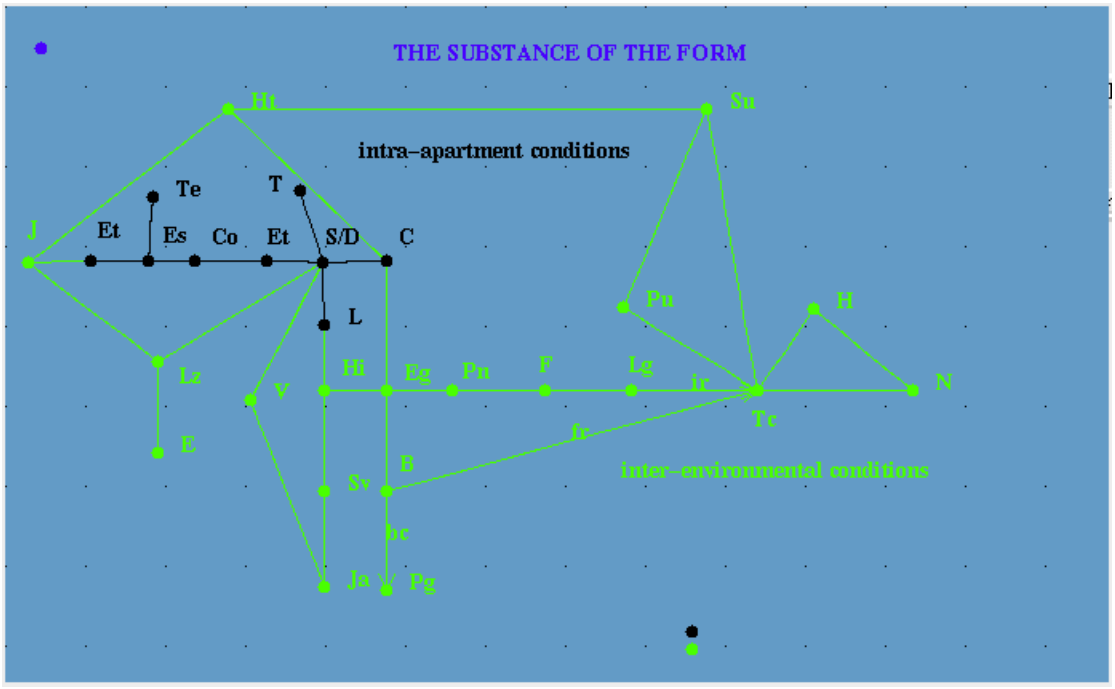


Figure 7. This concern stresses the pluridimensional and topological nature of the architectural design. Its network of holistic relations. Special mechanisms must be introduced at the level of the software architecture to mirror this structuration geometrically.

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 – Hyginiezation 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

The description above shows clearly the intertwining of microworld and macroworld. Moreover the **MPSTW** (Figure 5) describes the structure of the microworld. The model generates the macroworld from this core, hence shaping a homo-iconic system: it consists of structures built from a single type of construct. All objects within the system have identical implicit semantics. Its sense of object is determined by a system of

Kim Sung-do (Sung-do, 2000) unravels its encompassing nature: beyond being human, it is a surface, a tissue, a textile. It is a crucible where the world, the body and the connections between the subjective and the objective are to be comprehended as surfaces-topologies; the skin occupies no depth, generalized tactile is found on the surface. In generalizing this hypothesis one can say that the tissue, the textile provides excellent models of cognition, excellent quasi-abstract, objects, initial varieties. It is an open-ended conception. Indeed topology is tactile due to physicality. The skin becomes a place of sensation, a map of cognition and phenomenology and so saying – a topological map.

What do I mean by this? I mean here in a first stage all the stakeholders interact. Gradually they may start simulating their know-how through multi-agent approach. However the aim here is to allow the urban ecosystem to unfold as a single organism, an autopoietic entity that is distributed in time and space by recursive partitioning into parts that are conceived similarly structurally speaking to tune in within the whole: Mother Earth, ecocontinents, bio-regions, ecocities, econeighborhoods, ecobuildings, etc. If built, these parts behave like autopoietic systems or machines mimicking the behaviour of human beings.

Hence the plot of the architecture conceived as a stage with many actors, suddenly may follow an approach where it is highly automated.

7. Conclusions

The separation of composition of concerns achieved in the domain model and its underlying geometric modeling must be reflected in the architectural level. To achieve this goal, I have sharpened terminology and conceptualization around the concepts exposed in Greimas's

semiotic project as well as I have been delving deeper in the symmetry groups of the plane, adding the subgroup relationships of the crystallographic to integrate concerns vertically. This step requires deeper notions of music theory. Architectural composition tunes well with the concept of musical chords. For example there should be integration of the architectonic layout and the structural plan. This is one architectonic chord. Environmental chords integrating acoustics, comfort thermal and daylighting are another. At the final stage of development, all "architectonic chords must be vertically integrated.

This is very much dependent on the creativity of the designer and the client's expectations. Visualizing the software object as a single language evidences architecture as the place of narrative structures and discoursive structures.

The subjective version of the prototype based programming language Self Us enable one to implement in a seamless process the ideas on the domain model, however this in theory because in practice Self is an experimental language in its infancy. Prototypes are fundamental due to the arbitrary nature of each architectonic object in all dimensions.

An interactive and collaborative programming language like Self is a must to realize discourse at the architectural level or the participation of the stakeholders.

A graphical editor called TilingMorph based on the Morphic of Self eases the learning process and stimulates it by free-hand sketch and initial conceptualization in all levels. This is 90% of the effort held in architectonic thinking.

The implementation in Self necessarily entails an eXtreme Programming approach. Jecel de Assumpção and I have been working in a Sun station relatively slow and shared with many users. Before starting the implementation of the graphic editor based in my ideas, we tested the movement of the mouse and its ability to actualize the drawing at real time. Obviously this could be an obstacle. First we created a small program testing specifically to verify the time response of Self. Once confirmed the stability of the machine, we discarded this program and we incorporated this idea in the TilingMorph.

It would be very important for semiotic reasons to simulate the learning process of each designer, because as Peirce remarks: *The mind of an author cannot be reduced to what goes on in the brain since the process of writing also comprises the external manual activity of using the medium of ink to produce the written word.* (Nöth, 2002).

Hence the importance of extending the prototypes with the multi-agent approach, although unfolded according to the Greimas's semiotics.

Although delegation has aroused lots of interest lately and also the concerns with aspects, however they are not developed enough to implement my ideas directly through this tool (Bardou, 1998). Especially because they tend to happen at compiler or run-time. This is not interactive and collaborative and does not tune with my ecodesign model.

Acknowledgments: Heartfelt thanks go to Lars Lundberg, Jan Bosch, Jim Coplien, Alistair Cockburn, Winfried Nöth, Daniel Bardou, FAPESP, João Antonio Zuffo who enabled me to write this paper.

References

[Bäumer, D.: Software-architecturen für die rahmenwerkbasierte Konstruktion grosser Anwendungssysteme. Informatik der Universität Hamburg. PHD Thesis. 2000

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

Bardou, D.,1998. Roles, subjects and aspects: do they relate? LNCS 1533. ECOOP'98 Workshop Reader

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

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

do XV SBES Simpósio Brasileiro de Engenharia de Software. 177-192.
Galal, G. and Mens, T. : ECOOP'01 Workshop Report. IV Workshop on OO Architectural Evolution. To appear Workshop Reader. Budapest. http://prog.vub.ac.be/OOAE/ECOOP2001/ECOOP2001-OOAE.HTML

Galal, G. Software architectonics: towards a research agenda. II Workshop on OO Architectural Evolution. ECOOP 200

Garcia, A. et al: An aspect-based approach for developing multi-agent object-oriented systems. p. 177-192 Anais do XV SBES Simpósio Brasileiro de Engenharia de Software. 03 a 05 de outubro de 2001. Rio de Janeiro. Brazil
Greimas, A.J. et Court, J. 1979 Sémiotique. Dictionnaire raisonné de la theorie du langage. Paris

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

http://groups.yahoo.com/group/self-interest/links
Kiczales, G. et al, 1997. Aspect-oriented programming. Proceedings of ECOOP'97. LNCS 1241, 220-242

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

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

Serres, M., 1986 Les Cinq sens. Philosophie des corps m/s T1 Paris. B. Grasset. 1986

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
Sung-do, K, 2001. The aesthetic phenomenology of abduction. http://www.rz.uni-frankfurt.de/~wirth/texte/sung.html

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