U.P.B. Sci. Bull., Series C, Vol. 68, No. 3, 2006

HIERARCHICAL INTELLIGENT RECONFIGURABLE SIMULATION

T. NICULIU, S. COTOFANA*

Suntem inteligenţi, deci suntem conştienţi. Suntem în evoluţie, deci construim. Nu suntem singuri, deci trebuie să contribuim. Inteligenţa = (adaptabilitate, conştienţă, intenţie) e complementară credinţei = (intuiţie, inspiraţie, imaginaţie). Conştiinţa = (conştienţă, inspiraţie) integrează complementarităţile atât structural prin componente, cât şi funcţional prin rezultate (intenţie, imaginaţie). Cercetarea inteligenţei, prin simularea ei pentru a simula inteligent, cere studiul structurilor abstracte esenţiale: mintea umană, diferitele tipuri de ierarhii, şi simularea ca relaţie între funcţie şi structură. Inteligenţă şi credinţa, ca orice altă dihotomie, pot converge împreună spre integrare, ori se pot distruge reciproc de nu sunt asociate prin conştiinţă.

We are intelligent, so we are conscient. We are in evolution, so we construct. We are not alone, so we have to contribute. Intelligence = (adaptability, intention, consciousness) is complementary to faith = (intuition, imagination, inspiration). Conscience = (consciousness, inspiration) integrates the complementary parts both structurally through the components, and functionally by the results (intention, imagination). Researching intelligence, by simulating it to simulate intelligently, demands the study of essential abstract structures: the human mind, the different hierarchy types, and the simulation as relation between function and structure. Intelligence and faith, as any other dichotomy, can converge together to integration, or can destroy each other if they are not associated by conscience.

Keywords: Conscience, Faith, Hierarchy, Intelligence, Simulation.

Introduction

Both intelligent simulation and the simulation of intelligence demand transcending the present limits of computability toward simulability by an intensive effort on extensive research to integrate essential mathematical and physical knowledge guided by philosophical goals. Reconfigurability is extended to the simulation itself.

* Prof., Faculty of Electronics, Telecommunications, and Information Technology, University POLITEHNICA of Bucharest, România; Prof., Faculty of Computer Science and Engineering, Delft University of Technology, The Netherlands

Faith and Intelligence are ☯ in our life (Way, Truth, Life)

First, by a self-aware simulation, we get self-control of the simulation process; therefore, we build a knowledge hierarchy corresponding to the simulation hierarchy. Then, by expressing both simulation and knowledge hierarchies in the reference system of the basic hierarchy types (classes, symbols, modules), we create the context for a self-organized simulation.

Simulation relates function to structure. Reality is not confined to Nature, as the cardinal of the discrete $\mathbb{N}$ is less than cardinal of the continuous $\mathbb{R}$, but Reason, which is natural, as its cardinal $|\mathbb{N}| = |\mathbb{Q}|$, is dense in the Reality, as $\mathbb{R}$ is the analytical closure of the (discrete) rationales. This suggests that neither pure reason-based adaptability nor pure intuition can approach Reality without being integrated by conscience and communicating by intention and imagination.

The reference system of the basic hierarchy types (classes, symbols, modules) is derived from the main partition of our Real Life (Beauty, Truth, Good). Therefore, we try to model the conscience to reach for intelligence simulation, and then to apply this to intelligent simulation.

Intelligence, (consciousness, adaptability, intention) and Faith, (inspiration, intuition, imagination) are complementary parts of the human mind, nondeterministic linked by Conscience, (consciousness, inspiration). The historical experiment of the pure reason had sense in our evolution but should have ended long ago. Human thoughts can not be explained or handled by adaptability-based reason, even if nondeterministic or parallel. Reason has to extend to intelligence in the context of faith. An intuitive way is to integrate consciousness, then intention to intelligence, and then to extend the research towards imagination inspiration and intuition.

The power of abstraction is the real measure for the human mind. Turning abstraction into comprehensive construction could be the aim of humanity, the unique God for different cultures of free humans.

We have to recall our conscience to reintegrate our mind and to remember that society has to assist humans to live among humans, not to consider that humans just have to work for the society. An operating system serves the autonomous programs, both for the function of the hard and for development of the soft. In the same way, the society has to assure health and education for everyone, and encourage search and research for every healthy and educated human. We have to reconfigure the society to get and stay reasonable, and to let the humanity search further for Reality.

Freedom is understood necessity

Georg Friedrich Wilhelm Hegel

2. Reconfiguration

Reconfiguration continues the ideas of hardware-software cosimulation, intending to extend the software flexibility to hardware, as parallel software tries to get closer to hardware performance. The experimented ways to reconfigurable design are Field-Programmable Gate Arrays for circuits [14] and reconfigurable networks for systems [11]. Our project extends the reconfigurability to the simulation itself. Towards a self-aware simulation to control the simulation process we build a knowledge hierarchy corresponding to the simulation hierarchy, then by expressing both simulation and knowledge hierarchies in the reference system of the basic hierarchy types we create the context for a self-organization of the simulation (H-diagram). The basic hierarchy types correspond to essential views in languages/ systems theory, being derived from the main partition of our real life.

Reconfigurable computing architectures complement the existing alternatives of spatial custom hardware and temporal processors, combining increased performance and density over processors with flexibility in application. Recursive reconfiguration of the simulation process, at any hierarchy level, is allowed by different strategies that alter one of the technique/ model/ method if one of the imposed properties is not fulfilled after applying a technique, using a model and suitable methods for evaluation and reconfiguration. The process repeats for the initial description or the one resulted from prior (insufficient) improvement. This calls for an intelligent control system that assists/ automates the reconfiguration. The techniques use hard-soft model templates, whose methods are recursively handling the different components in the system's description. Measurement functions control the continuation process of the reconfiguration, what suggested bringing reconfiguration in the context of software and hardware, as the strategies can be expressed object-oriented/ categorical and developed/ understood mathematically. Intelligent self-organization needs consciousness to control adaptability for reconfiguration. We try to reach this goal integrating hierarchical intelligent simulation to nanotechnology realization.

```
class ReconfigurableSimulation { ...
void reconf (Bool tech, Bool mod, Bool meth) {
  if (tech) {technique = selectTechnique (TechType techniques);
    if (mod) {model = technique.selectModel (ModelType models);
      if (meth) { method = model.selectMethod (MethType methods);
                 (tech, mod, meth) = simulation (
                                technique, model, method);
    }}}} ...};
```

Simulation = (representation, goal)

Representation is a 1-to-1 mapping from the universe of systems (objects of simulation) to a hierarchical universe of models; hence, a representation can be inverted. A model must permit knowledge and manipulation, so it has two complementary parts/ views: description and operation. In a formal approach models correspond to classes and specifications to instances.

Dear God, search from the Sky, and see and research this Vineyard, implanted by Your Right, and complete it (in eternity)
Pantocrator, on the interior cupola of orthodox churches

3. Hierarchical Simulation

The research on cosimulation inspires the study of essential abstract structures: human mind, different hierarchy types, and simulation - as relation between static and dynamic structures, or even, at a higher abstraction level, between structure and function [10]. Towards this goal we put in correspondence three triplets of concepts of different collaborating domains: hierarchy types (class, symbol, structure), simulation abstractions (syntax, semantics, pragmatics), basic philosophy: (Beauty, Truth, Good). More points of view confirm a selection of the essential items to begin marching on the true way (Fig. 1).

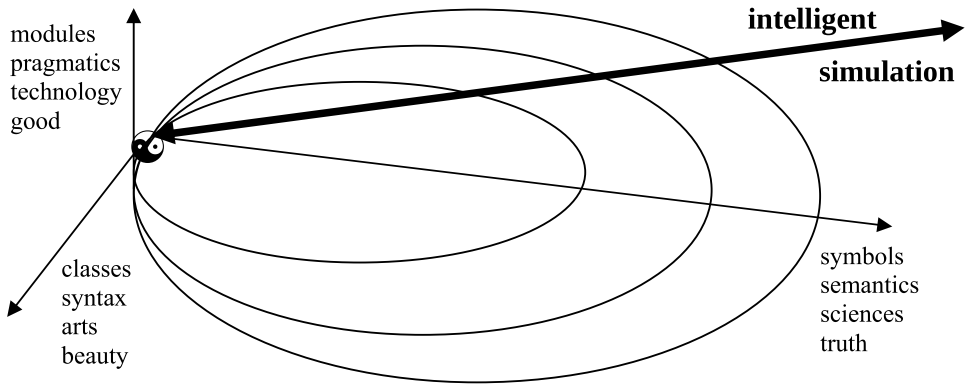

Fig. 1. H-diagram

The basic hierarchy types (classes, symbols, modules) correspond to (syntax, semantics, pragmatics) of the hierarchical language that has to express the intelligent simulation. Intelligent simulation results from the integration of the simulation hierarchy with its knowledge counterpart that represents a reflexive abstraction converging to self-consciousness of the intended adaptable simulation. The yin-yang represents the absolute functionality whereby the waves are increasingly structured hierarchy levels, both for simulation as for knowledge.

Knowledge and construction hierarchies cooperate to integrate (property-oriented) design, optimization, and verification into simulation; object-oriented concepts are symbolized to handle data and operations formally; structural representation of behavior manages its realization. A hierarchical approach is needed to handle both knowledge and metaknowledge. Hierarchy types open the way to simulate intelligence as adaptable consciousness by integrating the system and the metasystem. Hierarchy is the syntax of abstraction.

There are different kinds of abstraction; they need different types of hierarchy. Most abstractions are simplifying the approach, what is compulsory for complex object-systems. Classes abstract the form, symbols the contents, and partitions simplify the approach. All these enable the simulation hierarchy to assist construction, verification, optimization, and testing, being managed completely by pure reason, by discrete formalisms/ simulations. The natural limit of complexity is caused by the essentially sequential approach, whereby the real limit of computability results from the discreteness of our reason, when considered context-free in our mind. Understanding and construction should use correspondent hierarchy types, i.e., a reflexive kind of abstraction has to be expressed by the knowledge hierarchy type.

Metaphor is a popular instance of abstraction. God is the absolute abstraction; if we remember that *liberty is understood necessity*, we can detail the metaphorical thesis:

God is the evolution goal of our faithful intelligence

We can reduce abstraction to simplifying types (classes, symbols, modules, construction) hoping to approach the absolute liberty, i.e., considering God, the simplest item of the Reality. However, we can simulate/ construct/ live/ work associating a knowledge hierarchy to everything we do, aiming to understand constructively the most complex absolute necessity, defining God.

The power of abstraction is human's gift to surpass the natural limits by extending pure reason to real intelligence. As any other dichotomy pair, faith and intelligence can evolve convergent to integration, or can destroy one another if they are not linked together constructively. *Divide et Impera et Intellige* has three parts as *alle guten Dinge sind drei.* Mathematics develops from three basic structure types, usually integrating them: algebra, order, and topology. We divided our existence in three collaborating parts: arts, sciences, and technology, correspondent to our world of beauty-loving ideas, our world of truth-searching efforts, and our (presently exaggerated) world of good-aiming constructions.

Einstweilen, bis den Bau der Welt Philosophie zusammenhält, erhält sich das Getriebe durch Hunger, Furcht und Liebe

Friedrich Schiller

1. Mathematics (the most accessible art) discovers and studies types of structures: (algebra, topology, order) [2], correspondent to (construction, orientation, understanding) as example of correct and complete integration, to be followed by science and technology. *Art is for art*, so it is defining itself, looking for the Beauty.
2. Physics (the paradigmatic science) should integrate its fundamental forces in a theory [15], but also, as chapters, all natural and social sciences, leading them to really understand, apply, and inspire mathematics. Social sciences study a universe, as complex and nondeterministic as the natural one, so mathematics is at least as important to them as for natural sciences. Science raises the fear to more abstract domains, i.e., the research inspired by it can be defined hierarchically, as the *Fear of God* looking for the Truth.
3. Engineering has to be closely related to mathematical approach and integration of parts, not only to mathematical techniques, as to scientific courage and multiple views, not only to scientific results [5]. As reality contains the abstract ideas, even if physics could explain everything discretely, the power of continuum can not be forgotten, i.e., analog engineering should not be neglected in modeling and simulation. Paying attention only to the Good in our life, is most dangerous, as this part of the Reality, called *mental world* [13], defines its goal by its complement, so it is not better than this, if not closely constrained by Art and Science. Furthermore, different to art and science, there are two faces of the third approach, engineering and technology, which are not compulsory integrated. The correspondent sciences and technologies have to hold together, orienting engineers toward mathematical freethinking.

Das schöne wahre Gute

Johann Wolfgang von Goethe

is compulsory while we are evolving to God-alike humans

Hierarchy is a network that can represent any mathematical structure type (algebraic, topological, order). Hierarchies are leveled structures, which represent different domains. A level is an autonomous mathematical structure, containing abstract/ concrete entities, linked by level scoped relations. Abstraction relates the levels: this induces an order relation between levels, partial, concerning entities, and total, regarding the levels. Beyond the hierarchical point of view, the system can be formalized as an autonomous domain, structured by metahierarchical relations, building a level in a higher order hierarchical system. Hierarchical structures exhibit two complementary processing strategies: top-down and bottom-up. Coexistent interdependent hierarchies structure the universe of models for complex systems, e.g., hardware-software ones.

The hierarchical types belong to different hierarchy types, defined by simulation levels, classes, symbols, autonomous modules and knowledge abstractions. Abstraction and hierarchy are semantic and syntactical aspects of a unique fundamental concept, the most powerful tool in systematic knowledge; this concept is a particular form of *Divide et Impera et Intellige*; hierarchy results of formalizing abstraction. Hierarchies of different types correspond to the kind of abstraction they reflect ($\uparrow$ abstraction goal):

- Class hierarchy ($\uparrow$concepts) $\leftrightarrow$ virtual framework to represent any kind of hierarchy, based on form-contents, modularity, inheritance, polymorphism.
- Symbol hierarchy ($\uparrow$metaphors) $\leftrightarrow$ stepwise formalism for all kind of types, in particular also for hierarchy types.
- Structure hierarchy ($\uparrow$strategies) $\leftrightarrow$ stepwise managing of all (other hierarchy) types on different levels by recursive autonomous block decomposition.
- Construction hierarchy ($\uparrow$simulation) $\leftrightarrow$ simulation (design/ verification/ optimization/ testing) framework of autonomous levels for different abstraction grades of description.
- Knowledge hierarchy ($\uparrow$theories) $\leftrightarrow$ reflexive abstraction, aiming that each level has knowledge of its inferior levels, including itself. This hierarchy type offers a way to model conscience. The first idea is to consider/ remember that Reality is more than Nature, as the continuum of $\mathbb{R}$ is more powerful than the discrete universe of $\mathbb{N}$. The second analogy is that integer beauty is not enough to comprehend the Reality. The third argument is that reason is less than our real thoughts, as the cardinal of $\mathbb{Q}$ is $\aleph_0$. Actually, the knowledge hierarchy type can be called consciousness hierarchy type. $\mathbb{Q}$ is dense in $\mathbb{R}$, so pure reason could converge to reality, but the complexity problem limits the computability.

The classical activities in complex systems simulation that regard different levels of the construction or knowledge hierarchy, can be expressed symbolically then represented object-oriented and simulated structurally. Complex simulation needs consistent combination of mathematical domains and an intelligent compromise between consistence and completeness. Intelligence simulation implies a hierarchical approach of different types. Any application of it can be imagined as an educational system to discover models for conscience and understanding.

Constructive type theory permits formal specification and simulation, generating an object satisfying the specification [9]. The formalism for hierarchy types is the category theory [1]. The hierarchical types are objects of equivalent categories (functorial isomorphic) that formally represent hierarchy types. The consciousness hierarchy type communicates to the other hierarchy types by countervariant functors, while covariant ones connect the others.

The essential limit of the discrete computability, as of the computable intelligence, results from the self-reference, demanded by the integration of level and metalevel needed for consciousness. A hierarchical type is necessary to represent conscious knowledge. Even if for the moment other aspects can neither be constructive or intuitive, they should not be neglected.

For example, there are much more real things than those reasonably imagined, although between any two real numbers there is a rational one (not intuitive). And we know that if there is no cardinal between that of the countable sets and that of the continuous ones, then there exists no other logical value than true and false, what simply hurts the human in his love for nuances. This can be avoided only if we believe (not constructive) that an intermediary level between natural reason and Reality exists, as the wises think there are *angels* assisting humans to communicate with God (*Andrei Pleşu*, About Angels).

4. Hierarchical Cosimulation

Different domains permit a unified formalization in the theory of categories, and a unified representation using object-oriented templates. Simulation should remain correct, with extended requirements for the object-system, regarding complexity, optimization and (sequential/ parallel) competence for different domains. The hierarchical principle, applied to knowledge and simulation, (locally) bounds the complexity, by problem decomposition, and assures (almost) correct-by-construction design and efficient (design-adapted) verification. Cosimulation of coexistent domains is an important step for collaborative specialization, the next step to *Intellige* after *Divide et Impera*, and an essential need before approaching conscience modeling. Testability is the technological correspondent of sincerity, which is essential for intelligence and communication.

Hardware-Software Cosimulation

The hardware-software cosimulation of complex systems is imposed by the lack of compatibility or optimality associated with the initial hardware/ software partition of a design, and by the inefficiency of the design-verification cycle in the context of a fixed partition [7]. To unify simulation methodologies, we started from the results of different research directions: object-oriented hardware-software description, formal verification of software-hardware, automated synthesis of hardware systems. A unified representation for hardware and software allows techniques from one domain to be applied to the other domain. Therefore, a representation based on abstraction and object-orientation, used primarily for software, is employed for the hardware domain as well. In addition, existing software techniques, such as those used for verification of abstract data type implementations, can be used for hardware.

Knowing the features (mandatory: abstraction, hierarchy, encapsulation, modularity, message passing + optionally: typing, concurrence, persistence) that characterize an object-oriented language, they also make sense from the perspective of hardware modeling and simulation. Object-oriented specification of models can be based on general systems theory, what makes this approach applicable in all domains.

The designed framework permits self-organizing. It offers at any abstraction level of the simulation hierarchy: system description in a commonly used language extended for parallelism by synchronization items; automatic learning-based hardware/ software partition of the description; consistent communication between heterogeneous parts and with the exterior; simulation of the whole system during any design phase. Data abstraction can be used to represent hardware. A class corresponds to a set of elements with common static and dynamic characteristics. Thus, a hardware component can be treated as class containing state along with a collection of associated operations that can manipulate this state. E.g., at a higher level of abstraction, a processor is based on states, consisting of the values of the program counter and other internal registers, which is manipulated by its supported instructions. Starting with a collection of base classes, more specialized classes of components are derived through inheritance, e.g., the register class can be used to get special registers. A program counter (register with an increment operation), a stack pointer (register with increment and decrement), an instruction register (register in which the contents are divided into various fields), demand for additional member functions to extract the individual pieces of information from the register.

Generic types result from the ability to parameterize with types a software element, such as procedure or data type. This makes programs more general. The template concept, that realizes it in C++, can be applied to hardware components that act as containers, e.g., registers, register files. For example, a register can be viewed as a class with the operations read and write. The contents of the register correspond to its state, which can be accessed and manipulated using the operations read and write, respectively. Software engineering uses data decomposition to refine (derive implementations for) abstract data types. When modeled as data abstractions, hardware elements can also be refined using this decomposition technique.

Digital-Analog Cosimulation

The essential difference between analog and digital simulation paradigm is induced by that between the mathematical structures their models are based on algebraic for digital, analytical for analog. In view of intelligent simulation the whole intelligence has to be simulated, i.e., consciousness, intention, and adaptability.

The discrete parts of simulation, e.g., a sequence of decisions/ stimuli for simulation, do not easily match the continuity of the analog part [12]. Usually, the difficulty of analog simulation is avoided by defining an auxiliary representation domain, intermediate between the behavioral and the structural, where the problem is decomposed into topology selection and dimension computation. The first process is discrete and the second one is continuous over a restricted problem space. Object-oriented representation lends itself for this complementary form-content instance. However, topology selection would be more systematic if continuous modifications of the form were possible, and dimension computing is more efficient if symbolic algebraic methods are used. We searched the compromise between simulation algebra and analog analysis in three directions, all suited for an object-oriented Analog Hardware Description Language (AHDL):

1. defining upper levels of abstraction for the algebraic laws governed analog
2. modeling analog simulation in algebraic-analytical structure
3. association of analytical syntax to the analog simulation process.

Thermal-Analog Simulation

The development of CAD procedures for microsystems imposes the simulation of thermal phenomena as secondary effects to the main, analog (electronic, mechanic, optic, chemical) ones. As the microsystem components are modeled in an AHDL, the models can be enhanced with temperature dependence and power generation estimation. Moreover, models for environment and packaging conditions can be added as well. AHDL models permit direct simulation of the microscopic thermal transfer, and qualitative simulation-oriented representation of second order effects. Consequently, different physical domains, described by isomorphic analog laws, can be simulated in a unique representation [16]. Dynamics, circuit theory, hydrodynamics, thermodynamics, electrodynamics can be expressed with through-across concepts governed by dual topological laws for continuity and compatibility. AHDL enables a direct physical simulation of heat conduction, alternative to discrete heat equation: only the first order relation representing Fourier’s hypothesis is expressed in an AHDL model; its integration and discretization are realized by topological constraints that characterize AHDL structures. This suggests the idea that we follow towards formal verification: S*imulation is computer-oriented theory*

Behavioral Adaptable Design for Testability

Design-for-testability (DFT) must suit the behavioral specification of today’s complex system design. For intelligent simulation, it means *sincerity*. Referring to high-level synthesis DFT can operate before, while, or after it. The first choice permits the intervention of an intelligent agent for adapting the DFT technique, model, or method to the particular design.

We call it behavioral adaptable design for testability. It improves the testability, measured with adequate methods, directly on the behavioral specification or aided by special representations, that have to permit returning to the behavioral description after improving the testability of the system to be designed. It suits for hard, soft, or hard-soft systems.

Memory elements are represented in behavioral hardware descriptions by variables or signals. Variables are local description objects for processes/ subprograms, used to store intermediate values between sequential statements, characterized by free assignment. Signals are permanent description objects to link concurrent elements: components/ processes/ concurrent assignments, demanding synchronized assignment, declared locally - within architecture, block or other declarative region, or globally - in extended package. In the context of a process that is synchronized by a clock signal in a behavioral description, signals implicated in signal assignment generate memory during synthesis. An analogous rule can be formulated for variables: Inside a process, a variable that must hold values between iterations of the process implies memory elements. A variable that is set but not used between synchronization statements infers memory; a variable that is read before being assigned also infers memory. The context is not restrictive, as all concurrent statements are equivalent to processes (except direct instantiation). For called subprograms, the rules of memory inference can be deduced: pure functions do infer memory - while procedures do not.

The most used DFT techniques are Scanning, Built-In Self-Test, and Test Point Insertion. They can be applied at the different levels of the design hierarchy (behavior, RTL, logic) and can be combined. We began with Partial Scan applied to the autonomous blocks of the behavioral HDL specification, but the other techniques can contribute to improve the testability of the behavioral specification or the way to this goal. All types of hierarchies are implied in this approach: design abstraction levels, block structure, class framework, symbolization and knowledge hierarchies.

The Partial Scan problem is the selection of the scan registers following a strategy to find an optimum testability - complexity compromise; this is better nuanced by analog computing [4]. We combined Partial Scan methods to optimize the order to add memory elements to the scan chain, at behavioral level. An adaptable interface assures the translation, in both senses, from behavioral hard-soft description to a structural representation of the required behavior. The partial-scan selection uses a knowledge base to generate the weighted directed graph (flip-flops, combinational paths) and to return to text the differences caused by transformation for testability improvement. The rules of correspondence between description object (signal/ variable) assignments and registers, and those to translate the data flow in the behavioral specification to weighted arcs in the graph counterpart and to combine different testability measures in node weights, guide the first step. The second step is solved by incrementing rules for the hardware-software description. Partial-scan needs for the return translation a pointing scheme for the scanned objects among signals/ variables of the behavioral specification. This is managed by an adequate data structure in HDL. Flip-flops are selected for scan, but when a register is used in parallel, it candidates entirely for scan. The variables/ signals inferring memory are testability-related sorted to select incrementally the scan elements that will be mapped to the scan register.

5. Hierarchical Intelligent Simulation (ideas)

Applying *Divide et Impera et Intellige* to hierarchy types reveals their comprehensive constructive importance based on structural approach, symbolic meaning, object-oriented representation. Formal hierarchical descriptions contribute to a theoretical kernel for self-organizing systems. A way to begin is hierarchical simulation. A way to confirm is the object-oriented reconfigurable simulation. Essential relations are sketched before searching conscience models enabling intelligent simulation (Fig. 2).

Human = human (Humanity); human $\in$ Faith $\times$ Intelligence $\rightarrow$ Faith $\times$ Intelligence;
Humanity = (humans Set, evolution-oriented Structure).
evolution $\in$ (Hunger, Fear, Love) $\times$ (Technology, Science, Art) $\rightarrow$ (Technology,Science,Art).
Mathematics $\subset$ Arts = Human :: beauty-oriented activity (Science, Technology).
Physics = (natural $\cup$ social) Science = Human::truth-oriented activity (Arts, Technology).
Technology = Human :: good-oriented activity (Arts, Science).
simulation $\in$ Simulation $\subseteq$ Behavior $\times$ Structure $\Leftarrow$ Knowledge;
Knowledge $\Leftarrow$ Intelligence :: information();
Imagination $\Leftarrow$ | Intuition - Consciousness |; Intention $\Leftarrow$ | Inspiration - Adaptability |;
Adaptability $\Leftarrow$ simplifying_Abstraction (Imagination);
Consciousness $\Leftarrow$ reflexive_Abstraction (Intention);

Fig. 2. Class Human

...←Philosophy←...←human Culture←specific Knowledge←material Economics ← brute Force

The history of the common measure could be synthesized along the preceding line. The evolution of the common measure is conditioned by the conscient construction of intelligent agents to manage the lower stages, as industry enabled the mechanization of agriculture followed by the concentration on economics. The same scheme, or a more suited one, had to be applied long ago, changing (agriculture, mechanization, industry) by (pure reason, consciousness, intelligence) to (society, sincerity, humanity).

Napoleon Bonaparte and *Otto von Bismarck* started the reform, got convincing, and failed of unknown causes. Their names signify the recursive strategy used by the *pure reason experiment* pioneers (*René Descartes*, *Martin Luther*, and *Il Rinascimento*).

Conscience is self-awareness of individual faith and intelligence, as well as of the relation to the local context (society) and to the global one (Universe/ Reality) [6]. To appear it needed self-knowledge, what could have resulted from community conscience featured by an eternal human structure, e.g., from the past: shepherds, farmers, sailors, Africans, Amerindians, ... Each individual recognized himself in his cohabitants, being most adaptable and having a lot of intuition. The common measure evolution implies the construction of correspondingly intelligent agents to manage the lower stages and to concentrate on the higher ones. For example, industry enabled the mechanization in agriculture preparing for the concentration on economics.

Evolution is a multiple *Divide et Impera et Intellige* for conscience, associated to generating ($\rightarrow$) the *components* lacking of the mind at start, then assisted by them ($\swarrow$):

- individual-social-universal conscience $\rightarrow$ *inspiration* $\swarrow$
- space-time (structure-behavior) $\rightarrow$ *imagination* $\swarrow$
- discrete-continuous (natural-real) $\rightarrow$ *intention* $\swarrow$
- beauty-truth-good (art-science-technology).

The convergence process of evolution demands struggle against time, with structure as ally. Structure is sometimes too conservative, so it has to be reconfigured, at abstract levels, e.g., a plan, as at concrete ones. Conscience needs continuous feedback, not only discrete recurrence. Social and individual conscience are mostly divergent nowadays, i.e., we only performed *Divide et Impera*, neglecting *et Intellige*. It is high time to correct this!

Evidently, the anterior relations are oversimplified in order to move towards intelligent simulation. Although we claim they are intuitive and hope they are inspired, to begin, we neglect the essential but too primitive to understand intuition and inspiration, so (see further) formalizing the reflexive abstraction by the knowledge hierarchy type and the simplifying abstraction mainly by the simulation hierarchy type, it follows that:

Conscience = knowledge (simulation (Conscience)

This fixed-point relation suggests to model conscience by association of a knowledge level to any hierarchical level of the simulation process. To solve the fixed-point problem we build a metric space where knowledge°construction is a contraction - the elements implied in the construction get closer to one another in the formal understanding of the formal construct.

If, even in the sketch, we consider general functional relations between the essential parts of the faith-assisted intelligence, it results:

Conscience = knowledge (intention (Inspiration, simulation (imagination (Intuition, Conscience)

A generic modeling scheme defines the model universe as a mathematical theory or a design paradigm. Any entity has behavior (relations to other entities) and structure (internal relations). Behavior can be functional (context-free) or procedural (context-dependent). An algorithm is an entity that can be computer simulated, so it represents computability, behavior-oriented (understanding, verifying, learning)/ structure-oriented (construction, design, plan). The algorithmic approach is equivalent to the formal one: If a sentence of a formal system is true, an algorithm can confirm this. Reciprocally, for a verification algorithm of the mathematical sentences, a formal system can be defined, that holds for true the sentences in the set closure of the algorithm's results towards the operations of the considered logic.

David Hilbert's formal systems, *Kurt Gödel*'s construction algorithm, *Alonzo Church*'s λ-calculus, *Stephan Kleene*'s recursive functions, *Emil Post*'s combinatorial machines, *Alan Turing*'s machines, *Noam Chomsky*'s grammars, *Alexander A.Markov*'s normal algorithms, are the best-known (equivalent) formalisms for sequential reason-based computability [17]. Intelligence in evolution is the faculty to transform (analyze/ synthesize/ modify) abstract/ natural/ artificial objects, and representations, in the correspondent worlds of arts, sciences and technologies. Especially hierarchical reflexive: ideas about ideas, how to get to ideas, objects to transform objects, representations on representations, how to build/ understand representations. Evolution starts from the initial design of mental faculties for surviving of the whole system, and to the space-time context for communication between intelligent agents.

6. Hierarchical Intelligent Simulation (plans)

The alternative ways followed to extend the computability concept are suggested by approaches known from German literature, which is philosophy-oriented, trying to express essential ideas that link the conscious to the unconscious part of our mind. They respectively concentrate on the mental world of the good managed by technology, the physical world of the truth researched by science, and *Plato*'s ideal world of abstractions discovered by arts.

1. Faust (*Johann Wolfgang von Goethe*): heuristics - risking competence for performance, basing on imagination, confined to the mental world.
2. Das Glasperlenspiel (*Hermann Hesse*): unlimited natural parallelism - remaining at countable physical suggestions, so in the Nature.

3. Der Zauberberg (*Thomas Mann*): hierarchical self-referential knowledge - needing to conciliate the discrete structure of hierarchy with the continuous reaction, hoping to open the way to Reality [3].

Recurrence is confined to discrete worlds, while abstraction is not. This difference suggests searching for understanding based on mathematical structures that order algebra into topology. Recurrence of structures and operations enables approximate self-knowledge (with improved precision on the higher levels of knowledge hierarchies). A continuous model for hierarchy levels, without loosing the hierarchy attributes, would offer a better model for conscience and intelligence. A possible interpretation of knowledge hierarchies is: real time of the bottom levels - corresponding to primary knowledge/ behavior/ methods, is managed at upper levels - corresponding to concrete types/ strategies/ models, and abstracted on highest levels - corresponding to abstract types/ theories/ techniques.

Knowledge is based on morphisms that map the state-space of the object-system onto the internal representation of the simulator. An intelligent simulator learns generating and validating models of the object-system. Therefore: representation for design and verification should be common; the algebraic structures on which the different hierarchy types are based on should be extended to topological structures; the different simulation entities should be symbolic, having attributes as: type, domain, function. Knowledge-based approach separates representation from reasoning. A topology on the space of symbolic objects permits grouping items with common properties in classes. A dynamically object-oriented internal representation results, that can be adapted to the different hierarchy types. Topological concepts, as neighborhood, or concepts integrating mathematical structures, as closure, can be applied in verification and optimization, for objects as classes. The simulation environment prepares a framework for representing entities and relations of the system to be simulated, as general knowledge about the simulated universe.

Knowledge-based architecture, both at environment and simulation component level, ensures flexibility of the framework realization, by defining it precisely only in the neighborhood of solved cases. For representation, this principle offers the advantage of open modeling. The user describes models, following a general accepted paradigm that ensures syntactic correctness, leaving the meaning to be specified by user-defined semantic functions that control the simulation.

For example, a module in an unfinished design can be characterized by constraints regarding its interaction to other modules; the constraints system is a model, open to be interpreted, thus implemented, differently, adapting to criteria in a non-monotonic logic.

Mathematics contains structures that suggest to be used for self-referent models. The richest domain in this sense is functional analysis, which integrates algebra, topology, and order [8]: contractions and fixed points in metric spaces, reflexive normed vector spaces, inductive limits of locally convex spaces, self-adjoint operators of *Hilbert* spaces, invertible operators in *Banach* algebra.

7. Hierarchical Intelligent Simulation (example)

Let $(U, \{H_i \in S_h\})$ be a universe, structured by different hierarchies H_i and S_h the set of hierarchies defined on universe U: $H = (Rel_{eq}, \{(Level_j, Structure_j) \mid j \in S_l\}, Rel_{ord}, \{A_j \mid j \in S_l\})$ is a generic hierarchy, with: S_l the set of hierarchy levels, Rel_{eq} the equivalence relation generating the levels, $Structure_j$ the structure of level j, Rel_{ord} the (total) order relation defined on the set of hierarchy levels, $A_j \subset Level_{j-1} \times Level_j$, $j \in S_l$ the abstraction relation. U is a category, e.g., containing Hilbert spaces with almost everywhere-continuous functions as morphisms, enabling different ways to simulate self-awareness.

A hierarchical formal system can be defined (Fig. 3). Considering self-adjoint operators as higher-level objects of the knowledge hierarchy, these levels can approach self-knowledge in the context of knowledge about the inferior levels as of the current one, and having some qualitative knowing about the superior levels. Self-knowledge raises together with the abstraction of the hierarchy levels. The correspondence problem, i.e., associating the knowledge hierarchy to the simulation hierarchy, is managed by functorial morphisms over the various functors of the different hierarchies regarding the simulated system. To complete the simulation of the intelligence's components, intention is first determined by human-system dialogue.

1.	$(U, \{H_i \square S_h\})$, card(U) $>\aleph_0$	// hierarchical universe
2.	$\Sigma = F \cup L \cup A \cup K$	// functional objects
	$F = \{f \mid f : U^* \to U\}$	// global functions
	$L = \{f \mid f : Level_j^* \to Level_j\}$	// level structures
	$A = \{f \mid f : Level_j^* \to Level_{j+1}\}$	// abstractions
	$K = \{f \mid f: Level_j^* \times Level_{j+1} \to Level_{j+1}\}$	//knowledge abstractions
3.	$I = \Sigma^* \cap R$	// initial functions
4.	$R = \{r \mid r : \Sigma^* \times R^* \to \Sigma \times R\}$	// transformation rules.

Fig. 3. Hierarchical formal system

Further than modeling conscience to simulate intelligence we will be searching to comprehend inspiration, using *Lebesgue* measure on differentiable manifolds and non-separable *Hilbert* spaces. Perhaps even mathematics will have to develop more philosophy-oriented to approach intuition.

Conclusions

Society is only the memory of the past and the manager of the present problems to live together in respect of the others on the way to understand each other, by evolving toward more essential beings for an integrated existence. Conscience simulation demands transcending the present limits of computability. A way to begin is hierarchical analog-digital simulation. Applying *Divide et Impera et Intellige* to hierarchy types by the formalism of categories reveals their comprehensive constructive importance based on structural approach, symbolic meaning, object-oriented representation. Formalizing hierarchical descriptions, we create a theoretical kernel for self-organizing systems. Simulability is computability using the power of continuum. There are enough positive signs for this from analog electronics, control systems, mechatronics. Real progress towards this way of computation needs unrestricted mathematics, integrated physics, and thinking by analogies. Evolution implies the separation of faith and intelligence, so we have to better understand both, integrating them to human wisdom, to be divided further to get more human. Metaphorically phrased, our searches and researches should develop from the axioms:

1. God is unique 2. His ways are Uncountable 3. His plans are Hierarchical

Conscience is the link, in our mind, between what we are conscious of and what we are not. Presently, only the extended to Reason adaptability, and the unjustified Intention, are conscious. We can imagine an intelligent machine that looks like a human: robot <= labor, in Slavonic. It accumulates knowledge and behavior rules by preprocessing the senses, and it can change the interior defining rules (reconfigurable) corresponding to the behavioral (professional, ethical) knowledge that is considered most important, e.g., most recent or most decent. Therefore, it can consciously filter the actions that determine a new state of the context, what also means new knowledge to accumulate and to be conscious of (adaptability). It means, the dialog with the external environment determines the intentions. If the system had conscience, the external dialog would be more complex and interesting.

Consciousness only makes the adaptability more efficient, what, among others, transforms the human into the most powerful animal. Why do we compare the system without conscience with an animal, not to a human? It is true that we could compare it to an animal, if we had attributed intuition to it. However, what for should we do this, when the human just adapted to a consumption society? The built artificial objects and the socially useful natural objects send him the necessary messages to adapt consciously at the rising efficiency of the society. He neglects both the warnings from the superfluous Conscience and the unnecessary Intuition. If sometimes the two beasts shout too loudly, it is just unpleasant.

To be useful Intuition should be linked by Conscience to Intelligence, and intelligently bridled by Imagination. More, Intuition should also know to bridle by Intention the Adaptability. Whether he is human or animal, the human is anyway a machine, a social machine. His use is to contribute at the eternity, on an arbitrary level of evolution, of a materialistic consumption society. However, if the educated and encouraged consumption were not strictly materialistic, the human himself would escape from the vicious circle together with the others. More, the present level is artificial in the human evolution. The desire to stop the human evolution has no real argument. Nowadays, the evolution is forced to halt on an inhuman level, a consumption society transforming the society into a beehive without interest for Conscience and Faith, which most probably has been realized by destabilization of all revolutionary forms. The evolution is for the human among humans, assisted by a reasonably organized society that develops by the human, for the human towards the Human. We have to search and research for the aspects of the Reality, and of the human mind that reflects it, even if they are neither constructively nor intuitively expressible.

REFERENCES

1. *P.Ageron*, Limites inductives point par point dans les categories accessibles, in Journal on Theory and Applications of Categories, **vol. 7**, no.1, Jan. 2001, pp. 313-323.
2. *R Amoroso et al.,* eds., Science and the Primacy of Consciousness, Intimation of a 21st Century Revolution, Noetic Press, Orinda CA, 2000.
3. *M.Arbib*, ed.*,* Brain Theory and Neural Networks, MIT Press, Cambridge MA, 2003.
4. *L.Blum et al.,* Complexity and Real Computation, Springer Heidelberg NewYork, 1998.
5. *A.Ciaffaglione*, *P.diGianantonio*, Constructive Real Numbers, in Proceedings of TYPES*,* Lecture Notes in Computer Science, **vol. 2277**, 2001.
6. *D Hofstadter*, Gödel, Escher, Bach, Eternal Golden Braid, Vintage, Washington DC, 1979.
7. *K.Keutzer et al.*, Orthogonalization of Concerns and Platform-based System-Level Design, in IEEE T-CAD of Integrated Circuits and Systems, **vol. 19**, no.12, Dec. 2000, pp. 1523-1543.
8. *A.Kolmogorov*, *S.Fomin*, Analyse Fonctionnelle, MIR, Moskwa, 1977.
9. *J.vanLeeuwen*, ed., Handbook of Theoretical Computer Science, Elsevier, Amsterdam, 1990.
10. *C.Lupu*, Locality Measured by Contour Patterns. A Topographic Model, in Proc. of 15th IASTED Int. Conf. on Modelling and Simulation, Marina del Rey CA, 2004, pp. 50-54.
11. *R.Miller et al.,* Parallel Computations on Reconfigurable Meshes, in IEEE Transactions on Computers, **vol. 42**, no. 6, 1993, pp. 678-692.
12. *T.Niculiu*, *C.Lupu*, Categories for Hierarchical Simulation of Intelligence*,* in Proceedings of the IEEE International Conference on Computer as a Tool, Beograd, 2005, pp. 233-236.
13. *R.Penrose*, Shadows of the Mind, Consciousness and Computability, University Press, Oxford UK, 1994.
14. *J.Rabaey*, Reconfigurable Computing, in Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, München, 1997, pp. 127-132.
15. *J.F.Traub*, A Continuous Model of Computation, in Physics Today, **vol. 5**, no. 5, May 1999, pp. 39-43.
16. *B.Zeigler et al.*, Theory of Modeling and Simulation, Academic Press, Oxford UK, 2000.
17. *N.Zhong, K.Weihrauch,* Computability Theory of Generalized Functions, in Journal of Automated Computer Machinery, **vol. 50**, no. 6, June 2003, pp. 574-583.