Toward A Practical General Systems Methodological Theory

Nagib Callaos* and Belkis de Callaos**

Universidad Simón Bolívar, Apdo. Postal 89000, Caracas, Venezuela

*** ncallaos@usb.ve ** bsanchez@usb.ve**

ABSTRACT

Our main purpose in this paper is to describe the way in which we have been relating General System Theory (GST) to practice and to the design of a General Systems Methodology (GSM). Our first step was to apply GST to design a methodology for software development. Then, in a second step, by means of the experience/knowledge learned from applying the methodology to developing specific information systems, a continuous designing and re-designing process started, which simultaneously **generalized** the methodology and increased its complexity adding new methodical modules for an increasing diversity tasks needed for different **specific** systems and/or situations. The methodological kernel increased it **generality** and the sub-methodological modules increased in **specificity** and details. The methodological **intersection** of special methodologies increased its generality, because it its commonalities to a higher methodological diversity, and the **union set** of methodologies included more special methodologies for systems of different nature and for a higher diversity of situations, or environmental conditions. This paved the way for a General Systems Methodology which, because it includes cognitive/thinking methodological perspective it might take us back to the theoretical realm, i.e. to a methodological theory which, in turn, would pave the way to theoretical methodology. In this way Theory and methodologies would interact with each other in cybernetic loops, including negative and positive reciprocal feedback, as well as reciprocal feedforward.

1. INTRODUCTION

In the last 20 years we have been relating, in the context of action-research, GST to practice and vice versa. Our first step was to apply basic concepts and fundamental principles of GST to the domain of software development projects. We thought, and we still believe, that the so called software crisis is an **effectiveness** crisis, not an **efficiency** one, as it is usually approached. Then, we thought that the Systems Approach might provide concepts and principles with a high potential for improving this effectiveness, because the **pragmatic-teleological** truth of the this Approach is oriented to objectives attainment in praxis, and effectiveness is completely related to objective attainment. Some GST's concepts, like open system, adaptability, adaptability, emergence, integral control (feedback and feedforward), autopoiesis, Ashby's Requisite Variety, analogical thinking, end-means logic, pragmatic-teleological truth, Prigogine's dissipative structures, etc., helped us in re-designing our software development methodology in a way that significantly raised our effectiveness, and neared us to the systemic teleological truth. This re-design of our software development methodology raised our effectiveness, from about 10% to an average of 80%. As effectiveness measure we employed the most known one and the most easily understood by the users, i.e. the percentage of the number of lines of code (or function points) **delivered**, that went in **use**.

The methodology has been modified, (or re-designed) after each of its applications to a specific software development project, in the spirit of **action-research** and **action- learning**. After developing more than 120 software based information systems, we generalized the methodology in order to apply it to general information systems development projects; and after applying it to several projects of this kind we generalized it, one step further, in order to describe a systems methodology for designing, doing or acting; and from here we intended a Generalized System Methodology for thinking and acting, all of it in **an induction-deduction-production** cybernetic loops. With this attempt, we think we are back in the GST modifying and/or extending it by means of the practical effects learned from applying its methodological consequences to specific projects. In this way we combined action-research with action-learning, in an **intentional** context, to generate an approach that could be called **action-design**. This apprach is supporting our attempts to generalize the methodology by means of diversifying its specific applications and learning from this diversity. Doing so, we achieved practical and theoretical results. On the practical side, our statistics in the last 14 years shows an average of approximately 90% of effectiveness. On the theoretical side we arrived to a meta-methodology (on thinking and practice) that could be taken as a **methodological theory**, and it is supporting our intention to reach in the future to a **theoretical methodology**.

2. INFORMATION SYSTEMS DEVELOPMENT EFFECTIVENESS AND EFFICIENCY

We have already indicated that the so called *software crisis* is effectiveness, not an efficiency one, as it is assumed, explicitly or implicitly, by most authors, practitioners and methodologists. Indeed, some efficiency problems are rooted at a lack of

effectiveness. Solutions to both kinds of problems are not necessarily the same. On the contrary: in a dynamic context one kind of solution could oppose the other. Consequently, conceptual differentiation between both kinds of problems should explicitly be done by methodologists, while designing software development methodologies, and by practitioners and projects managers, while planning and controlling information systems development projects. Efficiency orientations are supported by positivistic epistemologies, which where identified by Iivari (1991; 1991b) as the kernel of present methodologies. Elsewhere (Callaos and Callaos, 1995) we presented general guidelines for an effectiveness orientation based on a non-positivistic epistemology, a systemic one, supported by GST and, specifically, by **Churchman's pragmatic teleological truth** (Churchman, 1971).

Effectiveness is a **praxiological** necessary antecedent to efficiency improvement. It could also be conceived as having epistemological and even ontological previousness. Meta-Software systems (code generator, CASE, etc.) speed software development up; it increases it development efficiency; but this has no benefit at all if the product does not get used by its users, and this requires it to be effective. **No meta-software can assure effectiveness.** It could help, supporting it. But to support something we should first have this *something*. So, **effectiveness is a necessary previous condition to efficiency** and to any supporting meta-software. Hence, it is evident the importance of a systemic, a pragmatic-teleological non-positivistic approach to information systems (IS) software development.

Effectiveness deficiencies in IS software development could be the effect of two possible causes: 1) to changes in user requirements in the software organizational environment or to changes in the environment of the developing **process**. The Standish Group research (Johnson, 1995) showed that 42% of the software projects are canceled because requirements were incorrectly collected and because of changes in requirements, in software environment and/or in the environment of the developing process. Consequently, the adaptiveness of the development process is very important - a must - in IS development methodologies. Software adaptiveness has been addressed for a long time (structured design, object orientation, etc.), but software's development process adaptability has seldom been addressed, until recently. The so called agile development (Extreme Programming, etc) is a movement toward process adaptability, which also requires product adaptability.

The product's efficiency orientation in software engineering is, in our opinion, a consequence of analogies drawn from the classical branches of engineering. In such branches, effectiveness has not been problematic, and process adaptiveness has not been required, because the physical laws does not change as users' requirements do, and the process' environment (which is mostly physical also) does not have the high level of uncertainty that the environment of the development process of IS software might have. So, analogies drawn from successful classical engineering practices and methodologies are not necessarily successful in IS development. Consequently, more adequate analogies, or metaphors, should be identified.

3. SYSTEMIC AND SYSTEMATIC METHODOLOGIES

After making a rigorous systemic definition of *definition* (Callaos, 1990) we defined *method*, *tool*, *technique and methodology*. These terms are sometimes used interchangeably, but we found out that there are significant conceptual differences among them. Elsewhere we presented these conceptual differences with an adequate level of rigor with its respective details (Callaos y Callaos, 1991; Callaos, 1996). Here, we will try to describe and highlight these conceptual differences by means of analogical thinking based on a metaphor: a road (method), a car (tool) and its driving (technique). There should be no confusions among these concepts, usually misused by most authors. A *method* is an intellectual and/or praxiological *road* which, as such, is a characterized by departure and arrival points. A *tool* is a means of transportation through one or more methods. *Technique* is the capability (aptitude and attitude) of handling a given tool in a given method.

Elsewhere (Callaos y Callaos, 1991; Callaos, 1996) we concluded that a *methodology* is a set of methods, tools and techniques related in an *a priori* way (i.e., **systematic,** closed methodologies), or in an *a posteriori* way (i.e. **systemic,** open methodologies). Systematic methodologies are supported by the metaphor of **mechanism** (closed, predetermined behavior, efficiency oriented and rigid), while the systemic ones could be conceived, by analogical thinking, as based on the **organism** metaphor (open, non-determined behavior, effectiveness oriented and flexible). At this point, it is evident how important would be the support of Bertalanffy's General Systems Theory (GST) for the design of a **systemic methodology** for IS software development.

Consequently, we applied several concepts and principles of GST to both: 1) the design of a systemic (open, flexible, adaptable, and effectiveness-oriented) methodology for Information Systems development, and 2) the design of a **systemic meta-methodology**, by means of which we designed and applied the referred IS methodology. Cybernetic recursiveness was applied, i.e. concepts and principles applied succeessfullt at the methodological level were also applied to the meta/methodological level, an viceversa, in cybernetic loops. In this way, and in the special case of software based information systems, we increased our methodological effectiveness (measured as the relation between the number of lines of code, or function points, in use and the number delivered to the user) from 5% to 12%, in the first attempt, and to 75%, after approximately 10 years of continuous re-design, through a combination of action-research, action-learning and action-design. In the last 10 years we even achieved approximately an average of 90% of effectiveness.

4. GENERALIZATION OF THE METHODOLOGY

The highly and doubtless increase of our methodological effectiveness in the domain of software-based IS development, encouraged us to apply our systemic methodology and meta-methodology, conceptually described elsewhere (Callaos and Callaos,1994a) to more complex situations and larger projects, where, software development was just one subsystem, and non-software subsystems (as organizational procedures, planning

processes, etc) were also present. In such cases, we observed also an increased level of effectiveness. So, we attempted to apply the methodology to non-software-based systems, as for example: design of training programs, design of the Latin American School for Executives and Statesmen (Callaos and Callaos, 1992), organizational design, national development design (Callaos, 1995b), etc. The effectiveness in such cases is not as high as in the IS software development, but it is increasing.

It is in the spirit of such kind of generalization of the methodology, that we applied it recursively to the meta-methodological level. We will return to this point in section 9.

5. METHODOLOGICAL SYNERGISM

In the spirit of the GST, we used the **organism** metaphor as basic ingredient in the analogical thinking that supported the methodological design. Consequently, we identified several properties that a systemic methodology should have, most of which could be found in Bertalanffy's GST and other authors' GSTs: openness, autopoieses, flexibility, negentropy generation, elasticity, anticipation, evolution, etc. Most of these properties influenced our methodological design. Our objective has been to design a methodology that would have such properties or, at least, analogous properties, which would cause an increase in the effectiveness of the methodological system being synthesized.

The property of **evolution** was fundamental in our methodological design. First, the **biological evolution** metaphor was used, and then the **ontological evolution** supported our analogical thinking. Consequently, the **ontological emergentism** was taken as something to be **imitated**. Elsewhere (Callaos, 1995) we described the basic ideas of such an attempt of imitation. Here, we will make a very brief description of the essential points.

Imitation is associated with the analogical thinking that characterizes the Systems Approach. Imitation has been, since Pythagoras, Plato and Aristotle, related to *poiesis*, i.e. with **productive** activities, i.e. with art and technique. It has been also strongly associated with **productive imagination**, which - according Kant - combines our experience into a single connected whole; with our faculties of ordering, recreating and uniting; and with invention (Trade, 1907). Many of the activities called *imaginative* - Ryle (1979) says - are *mock-performances*. So, it makes sense to try to imitate the natural synergic **evolutionary** processes in order to design a methodology (with a basically organic nature) for systems **development**.

The notion of *emergent reality*, formulated by Lewes (1873), was developed by Samuel Alexander, C. Lloyd Morgan (1923), and other neorealist authors. They basically conceive reality appearing in different **levels** as an *emergent evolution*. They contrasted *emergence* with *result*. While emergence refers to properties of a given ontological level not inferable from those belonging to the lower one, resultant refers to those properties that are inferable. Emergence is a **new property**. From a lower level A, with properties a_1, a_2 and a_3 which existed alone at time t_1, a higher level B would emerge at time t_2 (through a process of diversification and complication) with properties b_1, b_2 and b_3, inferable from a_1, a_2 and a_3, and properties b_4 and b_5 not inferable from them. A and B could coexist at t_2 as an AB reality. In such a case, a property a_4 might emerge at the lower level A, due to the higher level B coexisting with it. So emergence happens in both directions: from lower to higher levels, and vice versa.

This kind of metaphysical *emergentism*, or **ontological synergism** is, in our opinion, a very good analog for the intellection of a **methodological synergism**, an adequate metaphor to support the imagination and the design of synergic methodological systems and processes. We applied the model of ontological synergism to design a systemic systems methodology (Callaos, 1992; 1994a), in the general domain, and an information systems development methodology (Callaos and Callaos, 1994b; 1994c), as a special case. Consequently, we differentiated among various emergent levels. We distinguished **data** processing systems from **information** systems (and from **knowledge**/expert systems), as well as from and **organizational** systems, analogously to the physical, human, and social ontological realities, respectively. **Hence information systems have emergent properties related to data processing systems, and organizational systems have emergent properties as related to information systems**. Consequently, methodologies that are effective for the design of data processing systems are not necessarily effective for the design of information systems, because the novel emergent properties of the these were not considered at the lower level of methodologies for data processing systems design. Likewise, methodologies that showed to be effective at the level of information system design are not necessarily effective at the organizational systems design level, for similar reasons. On the other hand, new, emergent methods could be necessary to be included in data processing systems and/or methodologies should coexist with information systems design methodologies. The higher level methodologies (the information systems design ones) could provoke the inclusion of emergent methods in lower level methodologies (those oriented to data processing systems design and implementation). Prototyping methods are examples of emergent methods to be followed with the structured methods when a data processing system is being designed in the context of an information system, i.e. when data processing systems methodologies are used in the context of information systems methodologies. Likewise, emergent methods might be necessary to include them at the level of information systems design methodologies, if these are used in the context of organizational systems design methodologies. For example, reverse engineering methods at the software level might be included as a result of using business process re-engineering methodologies at the organizational level.

6. CO-EVOLUTIVE DEVELOPMENT

As we said, biological evolution was also an input metaphor to our analogical thinking. A systemic notion of such an evolution would combines, in a dynamic whole, **Darwinian** and **Lamarckian** perspectives (Callaos, 1995). These perspectives are, in our opinion, not contradictories, as frequently are taken, but polarities that require and necessitate each other for a comprehensive understanding of the phenomena of evolution. Conceptually, they maintain reciprocal relations, and form a systemic notion of evolution where mutations, **intrinsic** to the evolving system **propel** the evolutionary process, and **extrinsic** changes, in the system's environment, **provoke, pro-evoke,** the

system to adapt. By analogical thinking, we can associate these two complementary perspectives to what is known in the literature of *technological development and innovation* (TDI) as ***technological-push*** and ***demand-pull***. For a period of time there was a controversy about these two theories of TDI, but at the present, there is a consensus (still non-unanimous yet) that both forces are present (simultaneously or at different times) in the TDI. Likewise, thinking analogically, we could hypothesize that ***push/pull*** forces combine to **propel** and to **direct**, conjointly, the evolutionary process. We are conscious that the analogical thinking supporting the hypothesis formulation is not an epistemological guarantee, but our experience showed us that our *push/pull* (or Darwinian/Lamarckian) hypothesis, has in fact a **praxiological value**. The possibility of **epistemological value** might be addressed in the future, but from the methodological perspective of this paper, the praxiological value would suffice.

The conjoined push/pull evolution characterizes, not just the system, but also its environment, what we can call its co-system. This is because in the adaptive process, the system **re-acts** modifying itself, and **pro-acts** modifying its environment or its co-system. As consequence of the system's pro-action, different parts of the co-system may behave in three different ways:

(1) Some parts could have no reaction or pro-action at all, these could be called the **passive** parts; but

(2) The **active** parts could react adapting themselves to the system pro-action; and

(3) They could in turn pro-act modifying the system back.

The system in turn could behave in the same three ways, and so on goes the process in a **co-evolutionary** loop. **The system's evolution is cause and effect of its co-system's (environment's) evolution, and vice-versa** (Callaos, 1995).

Consequently, we have two co-evolutionary loops:

(1) Darwinian and Lamarckian evolutionary forces interact with each other, and

(2) System and environment adapt to each other, modifying themselves reciprocally.

This notion of double co-evolution has been of a high practical value. The idea of action-design emerged as a methodological consequence of it.

7. ACTION-DESIGN

One important lesson we learned from the notion of co-evolution, we briefly described above, was validated in practice. Knowledge and experience converged in the same conclusion: when we use a given methodology in a given project, we are actually inserting actively the methodological system in socio-technical environment. So, such a methodological system is acting (pro-actively) on its environment, which is not completely passive. Social environments react/pro-act back almost always, and natural environments frequently do the same thing, even if sometimes there is a time delay produced by ecological positive feedback loops that require time to grow and to generate noticeable effects. So, any methodology that we apply for systems synthesis is in fact imbedded in co-evolutionary process. Therefore, **an explicit formulation** of this characteristic should be done in order to make the methodology more effective and more efficient. Co-Evolution should be unambiguously specified in planning/implementing processes of action-design. A fairly good format for assembling the data to be entered in the action-design process, and for supporting the participation of designers, end users and clients in such a process, is a three-dimensional matrix (a cube) of system's reaction/pro-action vs. environment reaction/pro-action vs. past facts/future prevision-anticipation.

Elsewhere (Callaos and Callaos, 1993) we buckled down to present a systemic notion of *design*, and its relation to the concepts of *intention* and *action*. We concluded then that design is always intentional and action-oriented. The essence of design is to generate action in some direction and/or for some creation/production. It should not be isolated from action since it is strongly related to it. Both are parts of the same whole, both are members of the same organically dynamic system. **Design gives direction and action gives propulsion to the whole.** They are polar opposites, and, as such, they complement and require each other. So, there is no way of separating them without deteriorating their essence. Usually, design comes before and is input to material action. But when we are dealing with a complex system, **design and action should be conducted concurrently**, even tough design will initially get off alone up till an initial design of the first prototype, or archetype, of the wanted system is available. From there on, design and action should be interwoven, interacting with each other, by means of reciprocal loops of feedback and feedforward, in an evolutionary process. This process would be nurtured by the ingredients of **action-research** and **action-learning**, and could be named action/design. Action-research is a cybernetic/systemic method **of knowledge discovering**, where action support research, and vice-versa by means of reciprocal cybernetic loops of feedback and feedforward. In action-learning, action support **understanding** and **experiencing**, and vice-versa, by means of similar cybernetic/systemic loops. And, in action-design, action supports the **production/generation** of ideas, innovations and/or system synthesis, and vice/versa, by means of analogous cybernetic/systemic loops, and trough methods of action-research and action/learning. Here, action supports: 1) research, for discovering, 2) learning processes for understanding and experiencing, and 3) design, by means of the generation of new ideas and system synthesis for the solution of a specific problem or the satisfaction of existing, concrete requirements. Knowledge discovering and understanding are the **ends** of action-research and action learning, respectively, but they are **means** in action-design, which end is problem solving or system synthesis according objectives or given requirements.

8. INCREMENTAL CONJOINING OF DESIGN AND ACTION

When a methodology is being applied for a given system development and implementation, two kind of systems act/react/pro-act on its respective environments:

(1) The methodological system, i.e., the development **process**, and

(2) The system produced by such a process, the **product** being implemented.

Both systems have usually socio-technical environments which react/pro-act in a non-deterministic way. Consequently, a systemic methodology should be able to deal with its environmental uncertainty, and **methodological decisions should be taken under uncertainty** during the developmental process. Being this the case, it is not surprising that the **classical engineering** methodological approach (a highly systematic one) showed to be not adequate and with low effectiveness. We thought that a better approach would be the **managerial** one. **Descriptive** theories of decisions under uncertainty, in a managerial context, have been very helpful in our systemic methodological design. **Prescriptive** theories of decisions under uncertainty showed to be unhelpful in our case because they assume, as given, factors that are not given in our case; as, for example, the set of possible outcomes of the uncertain situation. On the other hand, descriptive theories usually make no assumptions not to be found in real world situations, because they are actually trying to describe how decisions are in fact being taken under real, non-theoretical, uncertainty. So, we reviewed this kind of theories and concluded that the Braybrooke and Lindblom's *disjointed incrementalism* (1970) is a suitable starting point for us.

According the *disjointed incrementalism* theory, managers and executives, when faced with uncertainty, they look for no optimal decision based on a global perspective, neither on an exhaustive list of present and possible future events. This would need more time than the executive is allowed to have before making the required decision. Furthermore, the content of the exhaustive list would very probably change before finishing the process of list identification. This would throw the executive into an infinite and sterile loop. Consequently, executives, when facing uncertainty, they take an incremental, short range, decision, according to a set of criteria; then, after executing the decision, they analyze its impact, verify their decision hypothesis, identify new variables in the dynamic environment, and review the criteria set, according which they take another incremental decision, and so on. Since the criteria and the data supporting the incremental decisions are changing in a non-foreseeable way, these decisions are disjointed.

But, a systemic methodology is a means for achieving specific pre-established objectives as, for example, a concrete system development, a particular design, a special problematic situation diagnosis or prognosis, etc. Then, we are in a position of decision making under uncertainty, but with concrete objectives to be achieved. Consequently, control means should be established such that the incremental process would converge in objectives' achievement, with and acceptable probability. Planned feedback and feedforward control loops could be integrated to the incremental process in order to direct it to the pre-established ends. In this way, incremental decisions will be no more disjointed, but **conjoined** in a process converging to an end envisioned in advance. A chain of alternative concatenated feedback and feedforward loops applied to the sequence of incremental decisions proved to be, in our experience, a good means for conjoining incremental decisions in a thinking/acting process, with a high probability of converging in the pre-established objectives. In this way thinking and acting are conducted in **parallel**, **simultaneously**, which are, as we conducted elsewhere (Callaos and Callaos, 1994a), an essential characteristic of a systemic methodology, and not in **series** as it usually is established in traditional projects management and engineering methodologies.

In the methodology, which basic guidelines we are trying to describe, after taking an incremental methodological decision, the implementation of this increment follows before the next incremental decisions is made. So, **project planning and execution are intertwined**. In the execution of each increment, clear and explicit negative feedback loops are designed and implemented in order to achieve the increment's micro-objectives. After completing the increment execution, and before deciding and planning the next increment, a feedforward control should be applied, according to the information and the experience gotten in the last increment's execution (action-learning), to the impact of the increment's execution on its environment, and to the possible changes that could have happened in such an environment and in the users' requirement. A next incremental decision will be taken based on the most proximate feedforward loop, and so on, in a conjoined incrementalism, i.e. in an incremental process converging on the pre-established ends.

In this way, incremental decisions are conjoined by explicitly planned control loops of feedback and feedforward, i.e. what we could call a **cybernetic conjoined factor**. But, experience shows that a **human conjoining factor** is also important, it is even necessary in many cases. System analysts/synthesists, system's end users and clients (those who take the final decisions on the projects restrictions: time, budget, etc.), should participate conjointly in feedforward control decisions. These decisions, unlike the feedback decisions, are not just technical, and should not be taken just by the analyst/synthesist and/or the project manager. They should be achieved by consensus with the end users and the client. In order to achieve this consensus, tradeoffs should be done among **technical**, **managerial** and **functional** variables, on one side, and between **efficiency** and **effectiveness** of the **product** to be achieved and the **process** used as a means. This take us to a three dimensional matrix containing 3x2x2 cells, among which tradeoffs should be done. Quantitative and/or qualitative operations research models could be combined with individual and/or collective decisions theory models (in ordinal or cardinal scales) in order to "*operationalize"* and to rationalize in a coherent procedure/process the achievement of the related tradeoffs. These tradeoffs are not to be done isolated from each other, they should be related systemically, in a **real total quality**, where *total* would not mean just **comprehensive** (as it is implicit or explicit in most total quality methodologies), but it would also mean **global**, i.e. systemic. This concept of strong (but not rigid) conjoining is a necessary condition if we are to apply incrementalism to system analysis and synthesis, and to project planning and implementation. Without it we will be at the risk of falling in Lindblom and Braybrook's disjointed incrementalism, which describe adequately the way executives take decisions under uncertainty, but it is non-adequate (taken alone) for systems analysis/synthesis (and project planning/implementation) under uncertainty. Elsewhere (Callaos y Callaos, 1994b) we made a detailed treatment of the **Systemic Total Quality** and the system of related tradeoffs we referred to here.

9. METHODOLOGY AND META-METHODOLOGY

We think that the methodology, we briefly outlined its general guidelines here, is fairly general as to be called a *General Systems Methodology*. We applied it to software development (more than 140 projects), information systems analysis and/or synthesis (more than 50 projects), executive support systems and decision support systems (13 projects), strategic planning of technological development and innovation (3 projects), educational systems (3 projects), etc.

Each one of these systems was designed and implemented with the conjoined incremental co-evolutionary action-design methodology roughly described here. At the same time, each one of the project represented, in turn, as a whole, an increment at the meta-methodological level, i.e. the meta-methodology by which we designed the methodology has also been, recursively, a conjoined incremental co-evolutionary action-design meta-methodology. Each one of the project was **one** incremental step at the meta-methodological level, and each project had **multiple** incremental steps at the methodological level. Both levels have been supporting each other. **The methodology is cause and effect of the meta-methodology, and vice-versa.** Both levels related in nonlinear cybernetic loops.

The particular project added new specificities and, hence, complexity to the methodological level, and more abstractness to the meta-methodological level. At the methodological level, the action involved in developing particular and concrete projects, had an increasing variety of **specific** tasks, which provided the required input at the meta-methodological level for increasing its **generalization**. So, the methodology is being continuously **more specific and diversified**, while the meta-methodology is being continuously more **general and integrated**. Thinking and acting has been dynamically related, with reciprocal dynamic nonlinear loops. At the thinking level **induction/generalization** and **deduction/ specification** have been interacting, with each other, through feedback (positive and negative) and feedforward loops. And both of them have been interacting (as a thinking whole) with action causing, and being caused, by the **production** process that has been generating working systems, with a continuously increased effectiveness. Induction processes provided **methodological hypothesis**, that were tested by means of deductive processes (which apply such hypothesis to specific projects) and the action required to produce the required working systems.

The process of relating the methodological and the meta-methodological levels, through cybernetic evolutionary loops (feedback and feedforward) of **inductive-deductive-productive** processes took us to the meta-meta methodological level and, hence, to a **methodological theory**, which has been treated lengthily elsewhere (Callaos, 1996), and it has been resumed in (Callaos, 1992). Since all this process is generating value added knowledge, we could attempt conceiving a **theoretical methodology** by means of changing the focus from the production of effective physical systems, to the production of effective cognitive/ epistemic systems. This would be a future research project.

10. REFERENCES

[1] Braybrooke, D. and Lindblom, C.E., 1970, **A Strategy of Decision: Policy Evaluation as Social Process**, New York: The Free Press.

[2] Callaos, N., 1990, "A Systemic definition of 'Definition'", (in Spanish), Universidad Simón Bolívar, Dpto. de Procesos y Sistemas, Caracas, Venezuela, 82 p., (a mimeograph).

[3] Callaos, N..1992, "A Systemic 'Systems Methodology'", **Proceedings of the 6th International Conference on Systems Research, Informatics and Cybernetics;** Baden-Baden, Germany, August 17-23.

[4] Callaos, N., 1993, "Ethical Considerations for Information and Knowledge Systems Development", in R. Packman (ed.): **Ethical Management of Science as a System**, Proceedings of the 37th Annual Meeting of the International Society for Systems Sciences, University of Western Sydney, Hawkesbury, Australia, July 5-9, pp. 265-274.

[5] Callaos, N, 1994a, "Conjoined Co-Evolutionary Action-Design", **7th International Conference on System Research, Informatics and Cybernetics;** Baden-Baden, Germany, August 8-11.

[6] Callaos, N., 1994b, "Meta-Strategic Formulation for Development", **The 2nd. Orwellian Symposium**, Carlsbad (Karlovy-Vary), Czech Republic, August 15-21.

[7] Callaos, N., 1995, "Significance of Synergetic Designing Methodologies", **The 8th International Symposium on Synergistic,** Bangkok, Thailand, January 13-20.

[8] Callaos, N., 1996, **A Systemic Methodology of Systems**, (in Spanish), Universidad Simón Bolívar, Dpto. de Procesos y Sistemas, Caracas, Venezuela.

[9] Callaos, N. and Callaos, B., 1991, "A Systemic definition of Methodology", in S.C. Holmberg and K. Samuelson (eds.), **Systems Science in the 21st Century; Integrating the New Sciences of Complexity in Service of Humans and their Environment**, Proceedings of the 35th Annual Meeting of the International Society for Systems Sciences; Ostersund, Sweden, July 14-20.

[10] Callaos, N. and Callaos, B., 1992, "Designing a Latin American School for Statesmen and Executives", **36th Annual Meeting of the International Society for Systems Science;** Denver, Colorado, USA, July 12-17.

[11] Callaos, N. and Callaos, B., 1993, "Design, Intention and Action**", 5th International Conference on Comprehensive Systems Design and Education**, International Systems Institute, Asilomar, Monterey, California.

[12] Callaos, N. and Callaos, B., 1994a, "A Systemic Methodological Support for Information Systems Analysis and Synthesis", in R. Trappl (ed.), **Cybernetics and Systems '94**, Vol. 1, World Scientific, Singapore, pp. 55-62.

[13] Callaos, N. and Callaos, B., 1994b "Designing with Systemic Total Quality", **Educational Technology**, Vol. 34, N° 1, 1994, pp. 29-36.

[14] Callaos, N. and Callaos, B., 1994c, "Conjoined Co-evolutionary Incrementalism for Information Systems development", in Zupancic, J. And Wrycza, S (eds.), **Proceedings on the Fourth Conference on Information Systems Development - ISD '94**, Bled, Slovenia, September 20-22.

[15] Callaos, N. and Callaos, B., 1995, "Information Systems development Effectiveness and Efficiency", **Proceedings on the 39th Annual Meeting of the International Society for Systems Sciences;** Amsterdam, Holland.

[16] Churchman, C. W., 1971, **The Design of Inquiring Systems**. New York: Basic Books Inc. Pub.

[17] Iivari, J., 1991a, **Paradigmatic Analysis of Contemporary Schools of IS Development. The Framework and Literature Analysis**. Department of Information Processing Science, Series A13, University of Oulu.

[18] Iivari, J, 1991b, "A Paradigmatic Analysis of Contemporary Schools of IS Development". **European Journal of Information Systems**, Vol.1, N° 4, 1991, pp 249-272.

[19] Johnson, J., 1995, "Chaos: The Dollar Drain of IT Projects Failures", **Application Development Trends**, January, pp. 41-46.

[20] Kant, I., **Critique of Pure Reason**; translated by N. Kemp Smith, London, 1929; and **Critique of Judgment**, translated by J.H. Bernard, New York, 1951.

[21] Lewes, G. H., 1873, **Problems of Life and Mind**, II, prop. V, Cap. III.

[22] Morgan, C. L., 1923, **Emergent Evolution**, I § 1.

[23] Ryle, G., 1979, **The Concept of Mind**; London.

[24] Trade, G., 1907, **Les Lois de l'imitation**, 1890; translated to Spanish: Las Leyes de la Imitación.