Formalized Analysis of Structural Characteristics of Large Complex Systems

Huaglory Tianfield

Abstract—Macrostructural modelization is paramount to the development of large complex systems (LCS). However, effective methodology has not been well established for macrostructural modelization of LCS. This paper explores the macrostructural modelization of LCS in terms of block diagram based model and grammar based model. Firstly, the macrostructural modelization problem of LCS is formulated. Secondly, a novel block diagram based model is proposed and established for LCS. Specifically, two novel general-purpose information-processing modules are proposed and constructed, called perception cube and decision spheroid. Through a distributed and nested structuring of the loops composed of perception cubes and decision spheroids, i.e., perception–decision links, an LCS is represented as a novel block diagram. Thirdly, a grammar based model is proposed and established for LCS through applying formal language theory to the block diagram based model. Specifically, perception cube and decision spheroid are visually represented as context-free grammars, named fusion grammar and synthesis grammar, respectively. Through a stratified constructive link between a stream of bottom–up growing fusion grammars and a stream of top–down growing synthesis grammars, a level of LCS is constructively defined and accordingly represented as a context-free grammar, named level grammar, for the first time. Then, a whole LCS is represented as a context-free grammar through a compounding of all level grammars. Finally, a case study on the incremental development of the computer integrated manufacturing system for a power-station boiler works is presented to demonstrate the potential usefulness of the proposed and established models of LCS.

Index Terms—Autonomous system, block diagram based model, distributed system, grammar based model, large complex system, macrostructural modelization, nested system, nesting, perception–decision link, system geometry, system modelization.

I. INTRODUCTION

Evidence for large complex systems (LCS) widely stand in practical applications. For instance, human brain–body systems, autonomous machine (such as machine–tool, grinder, vehicle, digger, transporter, loader, reactor, distillation-column, kiln, cracking furnace, especially robot) systems, man–machine combined (such as human–car driving) systems, autonomous urban traffic systems, autonomous production (manufacturing/industrial process) systems, and other diverse autonomous human/machine mixed systems. In fact, one of the most characteristic classes of LCS refers to either humans or engineering realizations of humans, or the systems in which humans or their engineering realizations are integrated as irreplaceable components. Significant trends emerging in systems and control in the last three decades [1]–[4] have represented some characteristics of such LCS to an extent. Because of the unconventional nature, LCS require new conceptualism and methodology [5]–[7].

Works that are closely relevant to LCS can be grouped into three topics, i.e., architecture, hybrid dynamics over continuous variates and discrete events, and performance learning.

Architecture involves synergic integration of a system. Literature [8] proposed an architecture for autonomous machines. It consists of three levels of top–down task decomposition, i.e., organization, coordination and execution. In terms of Jayes maximum entropy principle and the definition of knowledge rate, an analytical formulation was developed for the principle of overall structuring of levels in the architecture, i.e., increasing precision with decreasing intelligence [9]. More inspiring is that translation grammars [10], [11] and Petri net transducers [12], [13] were introduced for the modeling of levels, especially the coordination level in the architecture.

Literature [14], [15] proposed a reference architecture for a variety of LCS. It consists of three hierarchies of sensory processing, world model and task decomposition, respectively. Detailed description was made of spatial/temporal information integration and task decomposition, transformation performed by levels, and relationship between the more rapid and even timing at a lower level and the more slow and uneven timing at the higher level. This reference architecture was instantiated to autonomous robot systems [16], [17], multiple undersea vehicles [18], cooperative robot systems [19]. Literature [20], [21] proposed a three-level nested architecture, composed of planner, navigator and pilot, for autonomous robot systems, based on multiresolution planning and control. Besides, literature [22]–[25] studied reference models and architectures for computer integrated manufacturing systems. The mechanisms of multi-agent systems [26]–[29] are also found useful to the architecture problem of LCS.

Modeling and control of hybrid dynamic systems has become a focus in systems and control [30], [31]. A basic idea is to approximate hybrid dynamics over both continuous variates and discrete events into single-mode dynamics. That is, either continuous processes are abstracted into discretized features and events in terms of a partitioning of the entire state space (called event discretization), or hybrid dynamics is treated as a set of continuous processes which slide from one partition into another within the entire state space (i.e., partition-wise continuous processes). For event discretized systems, supervisory control of discrete event systems [32], [33] can be applied. For partition-wise continuous processes, the problem is converted as a
functional optimization subject to partition-wise continuous differential/difference equations.

It is necessary to note that as a result of multiple gradation in time drivers, the hybrid dynamics in LCS is much more complex than what is currently studied under the topic of hybrid dynamic systems. In fact, hybrid dynamics in LCS is not only that over continuous variates and discrete events, but more generally that over a spectrum of time drivers.

Evolution and learning in themselves can be viewed as a kind of approach for system design. On the other hand, only by means of analytical synthesis or optimization, as conventionally for continuous-variate servo/regulation problems, it is almost impossible that an LCS could successfully be designed by only one time. A feasible approach is in the first place to design an architecture for a system and then to refine the system behaviors under the architecture in an evolutionary way through performance learning, as in human problem solving. To drive the performance evolution of a whole LCS, it is realized that the learning must be capable of measuring in a unified way the different performances of different hierarchical levels. Literature [34] proposed using entropy as a common measurement for the performance of every level in the three-level architecture. Performance evolution, particularly for the organization level, was formulated as a stochastic approximation process [35]–[38]. The whole system was viewed as a hierarchical stochastic automaton [39], [40].

It goes without saying that macrostructural modelization is paramount to the development of LCS. However, effective methodology has not been well established for it. This paper will explore the macrostructural modelization of LCS in terms of two steps, i.e., construction of a novel block diagram based model and then formalization of it.

The remainder of this paper is arranged as follows. Section II formulates the macrostructural modelization problem of LCS. Section III proposes and establishes a novel block diagram based model for LCS. Section IV proposes and establishes a grammar based model for LCS through applying formal language theory to the block diagram based model. Section V presents a case study to demonstrate the potential usability of the proposed and established models of LCS. Finally, Section VI draws conclusions for the whole paper.

II. PROBLEM FORMULATION OF THE MACROSTRUCTURAL MODELIZATION OF LCS

A. Systems and System Development

A system, as a common sense, means an integration over a collection of components. A component can refer to either a physical entity or a logic set. The collection specifies a boundary of interest, which separates the system under consideration from its outside environment. This actually reflects a subjective scope or emphasis to deal with real-world systems.

While the definition of the boundary of the collection is subjective, perspectives of a system are objective. A system has multiple perspectives, as depicted in Fig. 1.

- functions as viewed from its external appearance;
- geometry as viewed from its internal framework;

Fig. 1. Multiple perspectives of a system.
the system and/or a rationalization of the computer supported system. Eventually, system development brings forth a target system, with operational infrastructure of either computer hardware/software or humans and most likely a mixture of both, as a better solution to the problem, as depicted in Fig. 4.

C. Macrostructural Modelization Problem of LCS

Virtually, it is not true that all perspectives of an LCS lie at the same level. Rather, there is a rough hierarchy among perspectives of an LCS, as depicted in Fig. 5. The transformation horizontally carried out by system development is vertically intersected by the multiple perspectives of the system.

In the transformation from existing system to target system, not all are varied, but there is some invariance. While operational infrastructure and the infrastructure dependent behaviors and information flows of the system are transformed from human and/or poor computer support to better computer support, others, like system geometry, and part of information flows and behaviors, are almost independent of operational infrastructures (whether human or computer supported) of and remain invariant over existing and target systems, as depicted in Fig. 5. This invariance is where macrostructural modelization makes sense.

Among the multiple perspectives of an LCS, macrostructural modelization of LCS involves geometry perspective, and partially information flow and behavior perspectives. How are subsystems of an LCS integrated into a synergic whole? What are the mechanisms of information flowing in LCS? All these problems are in the scope of macrostructural modelization of LCS. By means of a proper macrostructural model, various subsystems of an LCS can hold synergic integration in the processes where the LCS autonomously and dynamically performs specified tasks in a hostile environment under spatial, temporal and physical constraints.

The invariance over the transformation from existing system to target system is extraordinarily significant to system development. Thus, macrostructural modelization of LCS is absolutely necessary and is also the only feasible and effective start from which development of LCS could proceed. For simple systems, straight coming to the design of minute algorithms, without any macrostructural modelization in the first place, can often work because the corresponding macrostructural modelization problem, i.e., making choice between open-loop (feed-forward), closed-loop (feedback) via forward/backward channel controllers, and combined configuration, does not need much sophistication. However, for LCS, because of the complexity of system development, it is technically infeasible to straight proceed with minute algorithms at the very beginning of system development. Before any minute algorithms therein could proceed to be designed, there has to be a macrostructural modelization in the first place.

Thoughts implied by macrostructural modelization are well fitted with the general procedure by which humans solve complex problems, i.e., first grasping macro structure/logic of problem solving, and then being immersed in minute solving processes, which are governed by the macro structure/logic of problem solving. This recognizes the human/machine unification epistemology of LCS.

There are a variety of roles that macrostructural modelization can play in the development of LCS, typically as follows.

- macro logic/characterization;
- synergic integration;
- reference model/blueprint;
- framework;
- working procedure;
- specification;
- protocol;
- standardization;
- visualization;
- comprehensibility and so forth.

In such a sense, macrostructural modelization already turns out to be a great part of system development, particularly system analysis and system design.

D. Managing the Complexity of System Development

System development has much more than a system itself, because system development elaborates the growth of the system as well. In fact, complexity of system development can be attributed to two clustered dimensions, i.e., system geometry and system life cycle, as depicted in Fig. 6.

Reduction is always an effective approach to managing complexity. Given the above two clustered dimensions of system development, there are accordingly two ways of complexity reduction for system development, i.e., reduction along system geometry and reduction along system life cycle. Since risk, complexity and cost of system development are influenced much more decisively by system geometry than by system life cycle, system geometry reduction should in all cases be taken as the first alternative, and then system life cycle reduction as the second. After a whole system is decomposed along its geometry, risk, complexity and cost of system development will be exponentially reduced. Therefore, a meta-approach to system development can be put forward as follows.

Step 1) Overall requirement elicitation and development planning. This is upon a whole system and should also include decomposability analysis of a whole system, which will provide a guarantee for the integration of subsystem developments. If a whole system is decomposable, go to Step 2. Otherwise a
whole system has to be treated as a nondecomposable whole, and thus go to Step 4.

Step 2) Determination of a range to which system geometry reduction is applied on the development process of a whole system, as depicted in Fig. 7.

Step 3) Determination of a paradigm for system geometry reduction, i.e., incremental paradigm or parallel paradigm, according to the degree of decomposability of a whole system.

This is actually to determine arraying relationships among subsystem developments, which have the following possibilities.

- One after another, i.e., conservative incremental paradigm for weak coupling among subsystems.
- Overlapping, i.e., aggressive incremental paradigm for weak coupling among subsystems.
- Start aligned, i.e., parallel paradigm for loose coupling among subsystems.

Step 4) System life cycle reduction. If a whole system is decomposable, system life cycle reduction is a further reduction after system geometry reduction. Otherwise, if a whole system is nondecomposable, it simply has to be viewed as a nondecomposable whole, and only the system life cycle reduction is applicable.

Step 5) Integration of subsystem developments. This is an important part of the realization and test stage of the whole system development. The overall development planning (including decomposability analysis) upon a whole system is assumed to provide a guarantee for the integration of subsystem developments.

Macrostructural modelization of a whole system is useful and necessary for overall development planning (including decomposability analysis) and system geometry reduction upon a whole system, and for the communication, coordination, synchronization and integration among subsystem developments.
III. NOVEL BLOCK DIAGRAM BASED MODEL OF LCS

A. Basic Concepts

Definition 1: Right-to-left substitution (→). OR (⋅), and series (Ω). Leftward arrow (←) denotes mechanical substitution of a symbol on its left-hand side by that on its right-hand side. Semicolon (;) means OR. Notation Ω; is for abbreviating a series. \( x_{1\cdot\cdot\cdot \cdot n} \) is denoted as \( \Omega_{x_{1\cdot\cdot\cdot \cdot n}} \). Define \( \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \) if \( x_{1\cdot\cdot\cdot \cdot n} \neq \emptyset \) or \( 2 \) \( x_{\emptyset} = y_{\emptyset}, \emptyset \neq \min \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \) if \( y_{\emptyset} = \min \Omega_{y_{1\cdot\cdot\cdot \cdot m}} \) and \( \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} = \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \), where \( \emptyset \) is empty set, \( \Omega \), the subscript index set of nonnegative integers for symbol \( \sim \) here, \( \sim \) \( x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m} \). It can be seen that notation \( \sim \) is commutative with other operators.

Definition 2: Worlds of an LCS. It is assumed that an LCS can be partitioned into \( H \) levels, denoted as \( L_h, h = 1, \ldots, H \). As convention, two pseudo levels are defined, i.e., the (physical) controlled plant along with plant sensors and the controlled environment along with environment sensors, denoted as \( L_0 \) and \( L_{-1} \), respectively. For \( \forall h = 1, \ldots, H \), \( \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \) is sweepingly called the world of \( L_h \) of an LCS, denoted as \( \sim W_h \).

Definition 3: World data (WD), subtask commands (STC), and generalized data (GD). \( \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \) data of worlds, such as raw data (trajectory), trajectory knot, feature, pattern, geometrical/topological data, list, table, local area map, wide area map, stereo map, symbol, predicate calculus (well-formed) formula, frame, semantic net, etc.

For physical meanings, \( \sim \) WD; STC; GD. respectively, where \( D(\sim) \) is the domain of discourse for symbol \( \sim \).

For instance, for \( \sim \) WD, while \( \sigma = 1, \forall i(i) \in WD \) is a positive integer, and \( \theta = 1 \), then \( \forall (\omega(\theta) \in D(WD(i(\sigma)))) \) is called a world data (which is reasonable without saying); while \( \sigma = 2 \), \( \forall i(1), i(\sigma) \in WD \) are positive integers and \( [i(1) \sim i(\sigma)] \) may be either 0 or 1 or 2... and \( \theta = 3 \), then for \( \forall \omega(1), \omega(2) \in D(WD(i(1))) \) and \( \forall \omega(\theta) \in D(WD(i(\sigma)))) \), \( \omega(1), \omega(2), \omega(\theta) \) is also called a world data (which is readily comprehended to be reasonable).

Definition 4: Generalized control problem, knowledge, and search engine. For \( \forall h = 1, \ldots, H \), the problem how to make world \( W_h \) operate properly to perform required subtask commands is called generalized control problem. Specifically, a generalized control problem refers to the problem how to acquire spatially/temporally distributed world data sequence from \( W_h \), and then how to, based on this sequence, synthesize spatially/temporally distributed subtask command sequence to \( W_h \) to perform required subtask commands. Knowledge is the strategy of solving generalized control problems, and search engine is the strategy of guiding the utilization of knowledge upon generalized data.

Definition 5: Upward and downward attributes. Upward attributes \( \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \), \( \sim \) intelligence, granularity of representation of knowledge and generalized data, abstractness, macroscopic property, importance, spatial/temporal intervals at which generalized data appear, ranges spanned by subgoals, randomness of the spatial/temporal intervals at which generalized data appear, capability of solving generalized control problems, capability of symbol processing and logic reasoning, faculty of memory, capability of tackling uncertainty and fuzziness, interest in, control of and interaction with the environment ...

Downward attributes \( \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \), \( \sim \) precision, resolution of representation of knowledge and generalized data, concreteness, minuteness, rate and density at which generalized data spatially/temporally appear, evenness of the spatial/temporal intervals at which generalized data appear, number of subprocesses, forgetting, capability of numeric computing, interest in, control of and interaction with the controlled plant.

Each level of LCS distinguishes itself from others in terms of the upward and downward attributes contained in the knowledge and generalized data involved at the level.

B. Perception Cube and Decision Spheroid

Definition 6: \( \rho \)- and \( \varepsilon \)-processing. \( \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \), \( \sim \) spatial/temporal coordinate transformation, filtering, observation, estimation, learning, feature abstraction, pattern recognition, event detection, behavior discovering.

\( \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \), \( \sim \) spatial/temporal coordinate transformation, explanation, translation, approximation, quantization, interpolation, smoothing, refinement, and numeric error compensation.

For \( \forall \sigma \geq 1, \forall \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \in I_{\sim} \) and \( \forall \theta \geq 1 \), \( \forall \Omega_{x_{1\cdot\cdot\cdot \cdot n}, y_{1\cdot\cdot\cdot \cdot m}} \in \sim WD(\sim i(\sigma)) \) is sweepingly called \( [ \cdot ] \), here \( [ \cdot ] \) \( \sim \) \( \rho \)-processing; \( \varepsilon \)-processing while \( \sim \) \( \rho \)-processing, respectively.

For physical meanings, \( \rho \)-processing characterizes the core of perception process at a level to acquire world data, and \( \varepsilon \)-processing the core of the posterior processing at a level to make the synthesized preliminary subtask commands comprehensible, interpretable, acceptable, executable by the adjacent lower level.

Two novel general-purpose information-processing modules are proposed and constructed, called perception cube and decision spheroid, denoted as “Q” and “M”, respectively, as depicted in Fig. 8. The specific information processing they perform is as follows.

Each perception cube consists of two parts, i.e., world data acquirer and world data memory. Further composed of a search engine and a knowledge base, world data acquirer realizes \( \rho \)-processing of spatially/temporally distributed world data from a world. World data memory dynamically stores the acquired world data by a proper memorizing length and a forgetting factor. As viewed from its input and output, what a perception cube does is fuse low-level world data into high-level ones, with related high-level subtask commands taking part in the search engine for information fusion and memory organization of acquired world data.
Each decision spheroid also consists of two parts, i.e., subtask command synthesizer and posterior processing. The former further consists of a search engine and a knowledge base, and the latter realizes \( \epsilon \)-processing of the synthesized preliminary subtask commands. As viewed from its input and output, what a decision spheroid does is synthesize out low-level subtask commands based on the available high-level world data and under the high-level subtask commands.

Protocol between adjacent levels is realized jointly by the world data acquirer of the related perception cube and the posterior processing of the related decision spheroid.

C. Distributed and Nested Structuring of Perception–Decision Links

An LCS is the system on the vertical sections of which are nested levels and on the horizontal sections of which are the interconnections between bottom–up world data fusion process and top–down subtask command synthesis process. Perception–decision links constitute loops and these loops, with each of them as a whole, are distributed over spatial horizons and again nested level by level, as partially depicted in Fig. 9.

Therefore, through a distributed and nested structuring of the loops composed of perception cubes and decision spheroids, i.e., perception–decision links, an LCS can be represented as a novel block diagram, as depicted in Fig. 10.

As depicted by the block diagram based model, in an LCS, mainly involved are bottom–up spatial/temporal fusion [42] of world data, as jointly performed by all the perception cubes, and top–down spatial/temporal synthesis of subtask commands, as jointly performed by all the decision spheroids, more exactly, the interconnections between the two oppositely growing processes.

Literature [43] presented a unified framework based analysis on typical examples of LCS, i.e., human brain–body systems, autonomous robot systems, autonomous fed-batch reactor systems, human–car driving systems, autonomous urban traffic systems, and autonomous production (manufacturing/industrial process) systems. Considerably different from that in the design of numeric computational algorithms, what play a predominant role in the macrostructural modelization of LCS are a variety of heuristics, intuitions, experiences, etc. As a matter of fact, it is hard to demonstrate and evaluate a macrostructural modelization problem of LCS by a simple and immediate effort such as numerical computation, simulation or experiment, though the problem may be demonstrated and evaluated by long-term operations of LCS. Nevertheless, the unified framework based analysis [43] on typical examples of LCS visually illustrates the suitability and instantiatability of the proposed and established block diagram based model.

IV. GRAMMAR BASED MODEL OF LCS

The below will investigate the formalization [44, 45] of the above proposed and established block diagram based model through applying formal language theory.
A. Representation of Perception Cube and Decision Spheroid

Definition 7: Sorting of generalized data. It is assumed that the following occurs.

1) For non-negative real numbers $c_1 < c_2$, and $∀ R \subset \mathbb{U}_{i \in I_G} D(\mathbb{G}_i)$, there is a mapping

$$\lambda : \mathbb{D} \left( \frac{\alpha_i}{R} \right) \rightarrow [c_1, c_2]$$

where $\sim_i /R$ denotes attribute $\sim_i$ on $R$, $i \in I_R$, $∀ R_i \sim_i R_j$. Mapping $\lambda$ is called heuristic monotone, if for $∀ \alpha, \beta \in R$. $\lambda(\alpha) \geq \lambda(\beta)$ while heuristically speaking, $\alpha$ has more intensive attribute $\sim_i$ on $R$ than $\beta$.

2) There are two pieces of heuristic monotone mapping $\xi_i$ and $\eta_i$.

$$\xi_i : \mathbb{D} \left( \frac{\alpha_i}{R} \right) \rightarrow [0, 1], i \in I_\Lambda \setminus \mathbb{D}(\mathbb{G}_i)$$

$$\eta_i : \mathbb{D} \left( \frac{\alpha_i}{R} \right) \rightarrow [0, 1], i \in I_F.$$  

3) Generalized data $\mathbb{U}_{i \in I_R} D(\mathbb{G}_i)$ are sorted through a sorting process of a series of ordered and ordered sets $\mathbb{U}_{i \in I_R} R_i$ according to upward and downward attributes, which satisfies the following properties.

Property 1: Equal content but smaller base. $\mathbb{U}_{i \in I_R} R_i$ has the same content as, but smaller base than $\bigcup_{i \in I_G} D(\mathbb{G}_i)$, because the former is ordered but the latter not. That is

$$\bigcup_{i \in I_R} R_i = \bigcup_{i \in I_G} D(\mathbb{G}_i),$$

but $\#I_R \leq \#I_G$.  

Property 2: Disjoint element sets of $\mathbb{U}_{i \in I_R} R_i$.

$$R_{i_1} \cap R_{i_2} = \emptyset \text{ if } i_1 \neq i_2 \in I_R.$$  

Property 3: Orderliness of $\mathbb{U}_{i \in I_R} R_i$. For $∀ \gamma \in I_R$, $\mathbb{U}_{i \in I_R} R_i$ is an ascending order upward attributes $\mathbb{U}_{i \in I_R} R_i$, i.e.,

$$\sum_{i \in I_\Lambda} a_i \xi_i \left( \frac{\alpha_i}{R} \right) \geq \sum_{i \in I_F} a_i \xi_i \left( \frac{\alpha_i}{R} \right),$$

and descending order downward attributes $\mathbb{U}_{i \in I_F} \eta_i$, i.e.,

$$\sum_{i \in I_\Lambda} b_i \eta_i \left( \frac{\alpha_i}{R} \right) \leq \sum_{i \in I_F} b_i \eta_i \left( \frac{\alpha_i}{R} \right),$$

where $a_i$ and $b_i$ are nonnegative weighting coefficients satisfying $\sum_{i \in I_\Lambda} a_i = 1$, $\sum_{i \in I_F} b_i = 1$, and as convention, $\xi_i(\alpha_i/R_{t-1}) = +\infty$ and $\eta_i(\alpha_i/R_{t-1}) = -\infty$.

For any perception cube $Q(k)$, whether simple or complex, what it does is fuse spatially/temporally its input, a series of low-level world data $\mathbb{O}_{s=1}^{p(n+1)} \mathbb{O}_{t=1}^{q(n+1)} WD(k + 1, s, t) \in R_{\tau_{RQ}(\gamma)}/R_{\tau_{RQ}(\gamma)}$, into its output, a series of high-level world data $\Omega_{s=1}^{P(n+1)} \Omega_{t=1}^{Q(n+1)} WD(k, s, t) \in R_{\tau_{RQ}(\gamma)}$, where $s$ and $t$ are spatial and temporal indices, respectively, and $\tau_{RQ}(k - 1) < \tau_{RQ}(k) \in I_R$. This is depicted in Fig. 11(b). Generally speaking, $k(k - 1) \geq 0$, $k(k - 1) \leq t$, and the equal signs do not hold simultaneously for $s$ and $t$, meaning that fusion would always be done more or less. Moreover, in principle, there always exists a set of production rules, called fusion production, that can represent the fusion process.

Similarly, for any decision spheroid $M(n)$, whether simple or complex, what it does is synthesize output spatially/temporally its output, a series of low-level subtask commands $\mathbb{O}_{s=1}^{P(n+1)} \mathbb{O}_{t=1}^{Q(n+1)} SC(k, f, g) \in R_{\tau_{RQ}(\gamma)}$, based on a series of available high-level world data $\Omega_{s=1}^{P(n+1)} \Omega_{t=1}^{Q(n+1)} WD(k + 1, s, t) \in R_{\tau_{RQ}(\gamma)}$, and under its input, a series of high-level subtask commands $\mathbb{O}_{s=1}^{P(n+1)} \mathbb{O}_{t=1}^{Q(n+1)} SC(n + 1, f, g) \in R_{\tau_{RQ}(\gamma)}$, where $f$ and $g$ are also spatial and temporal indices as $s$ and $t$, respectively, and $\tau_{RQ}(n) < \tau_{RQ}(n + 1) \in I_R$. This is depicted in Fig. 11(a). Moreover, in principle, there always exists a set of production rules, called synthesis production, that can represent the synthesis process.

Definition 8: Series-symbol (●) and set of series-symbols (●). Notation (●) denotes viewing a set of generalized data as a symbol, e.g., $\bigcup_{i \in I} x_i$ is a series-symbol. Notation (●) denotes a set of series-symbols of finite length. Given a set $V$, then

$$\hat{V} = \left\{ x_{i_1}, \ldots, x_{i_n} \right\}, \quad 0 \leq n < \infty.$$  

Definition 9: Kindred (◇), α and β are called kindred, denoted as $\alpha \bowtie \beta$, if both $\alpha \sim \chi$ and $\beta \sim \chi$, $\chi \bowtie \chi \in \infty$.

Definition 10: Search engine intervened production. $\alpha \bowtie \beta$ expresses the production $\alpha \rightarrow \beta$ whose search engine is intervened (i.e., taken part in) by $\gamma$ under the condition that $\gamma \bowtie \beta$. Likewise, $\alpha \bowtie \beta$ expresses the production $\alpha \rightarrow \beta$ whose search engine is intervened (i.e., taken part in) by $\gamma$ under the condition that $\alpha \bowtie \gamma$.

Definition 11: Set of fusion production $F(k)$. It is assumed that $Q(k)$ have $u(k) > 0$ pieces of fusion production, and thus there are $u(k)$ fusion conditional formulas as follows:

$$f_{c_i} (\hat{x}) = f_{c_i} (\hat{x}) \left( \frac{s_{c_i} (k+1, \hat{x})}{\Omega_{s=1}^{P(n+1)} \Omega_{t=1}^{Q(n+1)} WD(k - 1, s, t)} \right)$$

where $w = 1, \ldots, u(k)$, $0 \leq \sim_{c_i} (k - 1, w) \leq \sim (k - 1, w)$. 

\[\sim \sim s, t.\]
The set of fusion production $F(k)$ can be expressed as (8), shown at the bottom of the page. In this definition, as convention, $\Omega_{\omega=1}^{\Omega_{\omega=1}} WD(k, s, t) \in \hat{R}_0$ express the raw data directly from plant and environment sensors.

Definition 12: Set of synthesis production $Y(n)$. The set of synthesis production $Y(n)$ can be expressed as (9), shown at the bottom of the page. In this definition, as convention, $\Omega_{k=1}^{\Omega_{k=1}} \Omega_{\omega=1}^{\Omega_{\omega=1}} STC(H + 1, f, g)$ express the total task commands specified.

Definition 13: The $j$th step fusion grammar $G_F(j)$. The $j$th step fusion grammar is a quaternion $G_F(j) = (N_F(j), T_F(j), P_F(j), S_F(j))$, where

1) $N_F(j)$: finite set of nonterminals

$$N_F(j) = \left\{ \left( \begin{array}{c} \omega \\ \omega \end{array} \right) \omega WD(j - 1, s, t) \right\}, \quad 0 \leq \omega_1 \leq \omega \leq \infty(j - 1), \omega \leftarrow s, t \right\} \cup \{S_F(j)\}. \quad (10)$$

2) $T_F(j)$: finite set of terminals

$$T_F(j) = \left\{ \left( \begin{array}{c} \omega \\ \omega \\ \omega \end{array} \right) \omega WD(j, s, t) \right\}, \quad 0 \leq \omega_1 \leq \omega_2 \leq \infty(j), \omega \leftarrow s, t \right\} \quad (11)$$

and $\exists \gamma \in I_F$ so that $T_F(j) \subset \hat{R}_0$. Obviously, $T_F(j) \cap N_F(j) = \emptyset$.

3) $P_F(j)$: finite set of production of form $\alpha \rightarrow \beta$

$$P_F(j) = F(j) \cup \left\{ S_F(j) \rightarrow \left( \begin{array}{c} \omega \\ \omega \\ \omega \end{array} \right) \omega WD(j - 1, s, t) \right\}, \quad 0 \leq \omega_1 \leq \omega_2 \leq \infty(j - 1), \omega \leftarrow s, t \right\}. \quad (12)$$

It can be shown that $\alpha \rightarrow \beta \in P_F(j)$ satisfies $\alpha \in N_F(j), \beta \in \left( N_F(j) \cup T_F(j) \right)^4$, where the superscript additive sign expresses the set of all strings of nonzero finite length.

4) $S_F(j)$: start symbol, $S_F(j) \in N_F(j)$.

Obviously, $G_F(j)$ is a context-free grammar. The language it generates $L_G(F(j)) \subset \hat{R}_0(\exists \gamma \in I_F)$, called the $j$th step fusion language, where the asterisk expresses the set of all strings of finite length.

In this definition, as convention,

$$L\left(G_F^{(0)}(j) \right) = \left\{ \left( \begin{array}{c} \omega \\ \omega \\ \omega \end{array} \right) \omega WD(0, s, t) \right\}, \quad 0 \leq \omega_1 \leq \omega_2 \leq \infty(0), \omega \leftarrow s, t \right\} \subset \hat{R}_0. \quad (13)$$

Definition 14: The $i$th step synthesis grammar $G_T(i)$. The $i$th step synthesis grammar is a quaternion $G_T(i) = (N_T(i), T_T(i), P_T(i), S_T(i))$, where

1) $N_T(i)$: finite set of nonterminals

$$N_T(i) = \left\{ \left( \begin{array}{c} \omega \\ \omega \end{array} \right) \omega STC(i + 1, f, g), \quad \left( \begin{array}{c} \omega \\ \omega \end{array} \right) \omega WD(i + 1, s, t) \right\}, \quad 0 \leq \omega_1 \leq \omega \leq \infty(i + 1), \omega \leftarrow f, g, s, t \right\} \cup \{S_T(i)\}. \quad (14)$$

$$F(k) \subset \left\{ \left( \begin{array}{c} \omega \\ \omega \end{array} \right) \omega WD(k - 1, s, t), \omega \leftarrow f, g, s, t \right\} \left( \begin{array}{c} \omega \\ \omega \end{array} \right) \omega WD(k, s, t) \left( \begin{array}{c} \omega \\ \omega \end{array} \right) \omega WD(k, s, t) \right\}, \quad \forall w \in \{1, \ldots, w(k)\}, f \in f_{kw} = \text{true}, 

\begin{align*}
&0 \leq \omega_1 \leq \omega_2 \leq \infty(k - 2 + i), \quad i = 1, 2, \omega \leftarrow s, t \\
&0 \leq \omega_1 \leq \omega_2 \leq \infty(k), \quad \omega \leftarrow f, g \\
&\omega \leq \omega_1 \leq \omega_2 \leq \omega_3, \quad \omega \leftarrow s, t
\end{align*}

and the equal signs do not hold simultaneously for $s$ and $t$. \quad (8)

$$Y(n) \subset \left\{ \left( \begin{array}{c} \omega \\ \omega \end{array} \right) \omega STC(n + 1, f, g), \omega \leftarrow f, g, s, t \right\}, \quad \left( \begin{array}{c} \omega \\ \omega \end{array} \right) \omega WD(n + 1, s, t) \right\}, \quad \forall w \in \{1, \ldots, w(n)\}, f \in f_{nw} = \text{true}, 

\begin{align*}
&0 \leq \omega_1 \leq \omega_2 \leq \infty(n - i + 2), \quad i = 1, 2, \omega \leftarrow f, g \\
&0 \leq \omega_1 \leq \omega_2 \leq \infty(n + 1), \omega \leftarrow f, g, s, t
\end{align*}

(9)
2) \( T_Y(i) \): finite set of terminals

\[
T_Y(i) = \left\{ \left( \bigcup_{j=1}^{n} \bigcup_{g=1}^{m} \text{STC}(i, f, g) \right) \mid 0 \leq \sim_1 \leq \sim_2 \leq \sim (i), \sim f; g \right\}
\]

and \( \exists r \in I_R \), so that \( T_Y(i) \subset \tilde{R}_r \). Obviously, \( T_Y(i) \cap N_Y(i) = \emptyset \).

3) \( P_Y(i) \): finite set of production of form \( \alpha \rightarrow \beta \)

\[
P_Y(i) = Y(i) \cup \left\{ S_Y(i) \rightarrow \left( \bigcup_{j=1}^{n} \bigcup_{g=1}^{m} \text{STC}(i+1, f, g), \right) \right. \\
\left. \quad \bigcup_{s=1, t=1}^{n} \text{WD}(i+1, s, t) \right\} \\
0 \leq \sim_1 \leq \sim_2 \leq \sim (i + 1), \sim f; g; s; t \}
\]

It can be seen that \( \alpha \rightarrow \beta \in P_Y(i) \) satisfies \( \alpha \in N_Y(i), \beta \in [N_Y(i) \cup T_Y(i)]^+ \).

4) \( S_Y(i) \): start symbol, \( S_Y(i) \in N_Y(i) \).

Obviously, \( G_Y(i) \) is a context-free grammar. The language it generates is \( L(G_Y(i)) \subset \tilde{R}_r(\exists r \in I_R) \), called the \( i \)th step synthesis language.

In this definition, as convention

\[
L(G_Y(H+1)) \supseteq L(G_Y(H+1))
\]

\[
\subseteq \left\{ \left( \bigcup_{j=1}^{n} \bigcup_{g=1}^{m} \text{STC}(H + 1, f, g) \right) \right. \\
\left. \quad \bigcup_{s=1, t=1}^{n} \text{WD}(H + 1, s, t) \right\} \\
0 \leq \sim_1 \leq \sim_2 \leq \sim (H + 1), \sim f; g; s; t \}
\]

(16)

**B. Constructive Definition and Representation of a Level of LCS**

**Process 1:** Stratified constructive linkup between a stream of bottom-up growing fusion grammars and a stream of top-down growing synthesis grammars.

Step 1) \( J(0) = 0, j = 1 \).

Step 2) If a \( j \)th fusion grammar \( G_F(j) \) can be (re)constructed so that it satisfies

1) Progression

\[
\exists J(j) \in I_R \text{ so that } L(G_F(j)) \subset \tilde{R}_j^k \text{ and } J(j) > J(j - 1)
\]

(18)

and 2) Full Adjacency (or Empty Nonadjacency), as depicted in Fig. 12(b)

\[
\left\{ \begin{array}{l}
\alpha \rightarrow \beta \in P_F(j), \alpha \in L\left(G_F^{(j)}\right) \neq \emptyset \\
\alpha \rightarrow \beta \in P_F(j), \alpha \in L\left(G_F^{(j+1)}\right) = \emptyset \\
\end{array} \right.
\]

(19)

then\n
\[
G_F^{(j)} = \left( N_F^{(j)}, T_F^{(j)}, P_F^{(j)}, S_F^{(j)} \right) \supseteq G_F(j) = (N_F(j), T_F(j), P_F(j), S_F(j))
\]

(20)

mark a backtracking point related to \( G_F^{(j)} \) for permanent use, and go to Step 3. Otherwise backward \( j \leftarrow j - 1 \) and turn to Step 2.

Step 3) If a linkup can be achieved at the top, as depicted in the top of Fig. 13, i.e.,

\[
L\left(G_F^{(j)}\right) \supseteq L\left(G_F^{(H+1)}\right)
\]

(21)

then \( H \leftarrow j, i \leftarrow H \) and go to Step 4. Otherwise, forward \( j \leftarrow j + 1 \) and turn to Step 2.

Step 4) If an \( i \)th step synthesis grammar \( G_Y(i) \) can be (re)constructed so that it satisfies
1) Full Adjacency (or Empty Nonadjacency), as depicted in Fig. 12(a)

\[
\begin{align*}
\text{Step } 4) &\quad \text{If } i = H, \text{ then erase all backtracking points related to } G_Y^{(H)}, \text{ and go to Step 6. Otherwise, go to Step 5.} \\
\text{Step } 5) &\quad \text{If } i = j + 1, \text{ then erase all backtracking points related to } G_Y^{(j)} \text{ and } G_Y^{(j-1)}, \text{ and turn to Step 1. Otherwise, backward } i \leftarrow i + 1 \text{ and turn to Step 4.} \\
\text{Step } 6) &\quad \text{If } i > 1, \text{ then forward } i \leftarrow i - 1 \text{ and turn to Step 4. Otherwise, succeed and stop.}
\end{align*}
\]

Up to here, a stratified linkup is constructively achieved successfully between a stream of bottom–up growing fusion grammars and a stream of top–down growing synthesis grammars. Accordingly, the stratified constructive linkup process stops. The whole process is further depicted in Fig. 14.

From the above process, a whole LCS is partitioned into \( H \) levels, and each level \( L_h \) consists of perception cube \( Q(h) \) and decision spheroid \( M(h) \). Levels of an LCS are fitted to a spectrum in terms of upward and downward attributes. The stream of bottom–up growing fusion grammars and the stream of top–down growing synthesis grammars between which a stratified constructive linkup is achieved, generate stratified kindred series-symbols of finite length, as depicted in Fig. 13. This can be formally stated by the following property.

**Property 4:** Stratified kindred series-symbols of finite length.

\[
J(h-1) < J(h) \in L_h, \quad h = 1, \ldots, H
\]

then

\[
L(\text{Step } 4) \cup L(\text{Step } 5) \cup L(\text{Step } 6) \subset \mathcal{R}_h, \quad h = 1, \ldots, H + 1
\]

This property is directly derived from Process 1 and is heuristically reasonable.

This property shows that a level of LCS is constructively defined through a stratified constructive linkup between a stream of bottom–up growing fusion grammars and a stream of top–down growing synthesis grammars. **Observation 1:** Level grammar \( G_L^{(h)} \) and its language

\[
L(\text{Step } 4)
\]

It is readily possible to construct a context-free grammar

\[
G_L^{(h)} = \left( N_L^{(h)}, T_L^{(h)}, I_L^{(h)}, S_L^{(h)} \right)
\]

where

\[
\begin{align*}
N_L^{(h)} &\equiv N_Y^{(h)} \cup \left( S_L^{(h)} \right) \\
T_L^{(h)} &\equiv T_Y^{(h)} \cup \left( S_L^{(h)} \right) \\
I_L^{(h)} &\equiv I_Y^{(h)} \cup \left( S_L^{(h)} \right)
\end{align*}
\]

and there is

\[
L(\text{Step } 5) \cup L(\text{Step } 6) \subset \mathcal{R}_h, \quad h = 1, \ldots, H + 1
\]

This observation follows directly from the known property that the union of a finite number of context-free grammars is also a context-free one. A level of LCS is represented as a context-free grammar, for the first time, through a compounding of a proper fusion grammar and a proper synthesis grammar, respectively out of the stream of bottom–up growing fusion grammars and the stream of top–down growing synthesis grammars between which a stratified constructive linkup is achieved.

This observation shows that the operation of the \( h \)th level of LCS is that given the world data

\[
L(\text{Step } 4)
\]

where \( h = 1, \ldots, H \), \( G_L^{(h)} \) is called the \( h \)th level grammar and

\[
L(\text{Step } 4) \subset \mathcal{R}_h, \quad h = 1, \ldots, H + 1
\]

This observation is presented from the \( h \)th level and the subtask commands

\[
L(\text{Step } 4) \subset \mathcal{R}_h, \quad h = 1, \ldots, H + 1
\]

This observation is presented from the \( h \)th level, the \( h \)th level grammar \( G_L^{(h)} \) generates spatially/temporally distributed higher-level world data.
C. Representation of a Whole LCS

**Observation 2:** Whole-system grammar \( G_{\text{LCS}} \) and its language \( L(G_{\text{LCS}}) \).

It is readily possible to construct a context-free grammar

\[ G_{\text{LCS}} = \langle N_{\text{LCS}}, T_{\text{LCS}}, P_{\text{LCS}}, S_{\text{LCS}} \rangle, \]

where

\[ N_{\text{LCS}} = \left\{ \bigcup_{h=1}^{H} X^{(h)} \right\} \cup \left\{ S_{\text{LCS}} \right\} \]

\[ T_{\text{LCS}} = \left\{ \bigcup_{h=1}^{H} \mathcal{T}^{(h)} \right\} \]

\[ P_{\text{LCS}} = \left\{ \bigcup_{h=1}^{H} \mathcal{P}^{(h)} \right\} \cup \left\{ S_{\text{LCS}} \rightarrow S_{L}^{(1)} \mid \cdots \mid S_{L}^{(H)} \right\} \]

and there is

\[ L(G_{\text{LCS}}) = \bigcup_{h=1}^{H} L(G_{L}^{(h)}) \]

\( G_{\text{LCS}} \) is called whole-system grammar and \( L(G_{\text{LCS}}) \subset \bigcup_{h=1}^{H} L(G_{L}^{(h)}) \) whole-system language. A whole LCS is represented as a context-free grammar through a combing of all level grammars.

This observation shows that the operation of a whole LCS is that starting from the data \( L(G_{L}^{(h)}) \) directly from plant and environment sensors and the specified total task commands \( L(G_{L}^{(H+1)}) \), whole-system grammar \( G_{\text{LCS}} \) generates various levels of spatially/temporally distributed world data and various levels of spatially/temporally distributed subtask commands. Whole-system language \( L(G_{\text{LCS}}) \) describes the series-symbols of finite length that are composed of spatially/temporally distributed world data and subtask commands, fused out and synthesized out by the stream of bottom–up growing fusion grammars and the stream of top–down growing synthesis grammars between which a stratified constructive link up is achieved.

V. CASE STUDY

A case study on the incremental development of the computer integrated manufacturing system (CIMS) for a power-station boiler works (PBW) is presented in the below to demonstrate the potential usability of the proposed and established models of LCS. Enterprise systems are nowadays becoming increasingly complex and as a result, the information processing required in enterprises has become much more complicated and difficult to handle than ever [46]–[48]. A CIMS is an enterprise-wide computer information management system, as depicted in Fig. 16.

An overall development planning was carried out for PBW-CIMS after requirement investigation, concluding that incremental paradigm be applied to the development of PBW-CIMS, as sketched in Fig. 17. Development phases are identified as follows.

**Phase 1** Build information infrastructure of PBW-CIMS to lay a foundation for enterprise-wide information integration. Specific tasks include the following:

- Set up and consummated standard and norm for data and information architecture for data and information all over the enterprise.
- To build an enterprise-wide computer network system, and to set up an intranet.
- To plan the structure of enterprise-wide databases, and to configure and implement a database management system.
- To set up bill of material (BOM) tables of typical power-station boiler products.

**Phase 2** Seize pressing-to-solve problems, rationalize enterprise-wide information flows, and expedite information processing within the enterprise.

Aimed at three bottleneck problems, i.e., cost control, material management and price quotation, the emphasis...
is placed upon financial subsystem, material supply subsystem and operation management subsystem of PBW-CIMS. Specifically

- reinforce costing mechanisms;
- raise instruments for material supply management.

In accordance with the functional requirements of PBW-CIMS upon material supply subsystem, through the acquisition and deployment of MRP-II (manufacturing resource planning) software system, set up computer information management system of material supply, to effectively lower the inventory of raw materials and the occupation of funds, and control the product cost;
- through the establishment of a product price quotation subsystem in operation management subsystem, expedite the process and increase the accuracy of product price quotation.

**Phase 3)** Based on the customization development of MRP-II software system, complete the information integration among financial subsystem, material supply subsystem and operation management subsystem. At the same time, set up basic variants of quality management subsystem, production management subsystem, technical design subsystem, and decision support subsystem.

In determining the precedence among the start and finish times of subsystem developments of PBW-CIMS, the following points should be taken into consideration.

1) Overall development planning of PBW-CIMS must lead first and all other development activities of PBW-CIMS must be posterior to its completion.
2) After overall development planning of PBW-CIMS is completed, it is recommended that standard and norm for data and information, networking engineering and database systems be immediately developed so as to quickly establish an infrastructural operating environment for PBW-CIMS.
3) It is recommended that financial subsystem, material supply subsystem and operation management subsystem be developed in Phase 2, and that the three subsystems be simultaneously developed to expedite the overall schedule of system development.
4) Quality management subsystem, production management subsystem, technical design subsystem and decision support subsystem can be developed in Phase 3. Likewise, it is recommended that the four subsystems be simultaneously developed to expedite the overall schedule of system development.
5) Later consummation project (e.g., workshop automation etc.) should be developed in the last.
6) Intervals between different development phases are made up by the enterprise according to practical necessity and circumstance.

**Benefits of incremental paradigm** are seen as follows.

1) CIMS is a very large and complex system, needing huge investments. Incremental paradigm effectively reduces risk, complexity and funds, and rationalizes both the schedule and the installment of investments.
2) Phase 1 of development is infrastructure building. Although completion of this phase does not seem to return immediate economic profits, information infrastructure must be built before all other subsystems could start, because information infrastructure is basis and common to all subsystems of the CIMS, and is of long-run influence and significance upon the CIMS and broadly upon the enterprise. In fact, the investment in Phase 1 is only once, and can eliminate potentially repeated investment later on with the project.

3) Phases 2 and 3 are arranged according to the pressing situation in which the enterprise is confronted with bottleneck problems. Completion of these phases will immediately solve the bottleneck problems of the enterprise and return radical profits in terms of organizational efficiency, productivity, cut cost, rationalized material and financial management, shortened time to market, freed potentials, and enhanced competitiveness.

Macrostructural modelization of the whole CIMS is useful and necessary for overall development planning (including decomposability analysis) and system geometry reduction upon the whole system, and for the communication, coordination, synchronization and integration among subsystem developments.

**VI. CONCLUSION**

By the invariance over the transformation from existing system to target system, macrostructural modelization is paramount to the development of LCS. This paper has explored the macrostructural modelization of LCS in terms of block diagram based model and grammar based model.

Perception cube and decision spheroid are generic and characterize two kinds of basic information processing universal in LCS. A perception cube and a decision spheroid constitute a closed loop. Such a loop, i.e., perception–decision link, has a good characterization of the elementary autonomy universal in LCS and is able to serve as an element to modelize LCS. A number of such loops are inherently distributed. A group of distributed loops (of relatively larger number) are nested into another group (of relatively smaller number). Such a distributed and nested structuring of the loops composed of perception cubes and decision spheroids, i.e., perception–decision links, visualizes the macrostructural patterns of LCS.

It is fairly straightforward to represent perception cube and decision spheroid as context-free grammars. For the first time, a level of LCS is constructively defined through a stratified constructive linkup between a stream of bottom–up growing fusion grammars and a stream of top–down growing synthesis grammars. Accordingly a level of LCS is represented as a context-free grammar, through a compounding of a proper fusion grammar and a proper synthesis grammar, respectively out of the stream of bottom–up growing fusion grammars and the stream of top-down growing synthesis grammars between which a stratified constructive linkup is achieved. This stratified constructive linkup process deeply visualizes the macrostructural patterns of LCS. Representing a whole LCS as a context-free grammar is through a compounding of all level grammars.
The two proposed and established models constitute a consortium of models for LCS. Between the two forms of models is not simple transformation from one form into another. The proposed and established block diagram based model is simple and readily comprehensible, and has uniformly unidirectional information flows throughout and extensive similarities over distributing and nesting. It is the block diagram based model that exclusively lays a feasibility for a formalization to be built on it. Actually the block diagram based model provides a basis of visualization so that grammar based model could be built up through applying formal language theory.

The proposed and established models are condensed, abstract, visualized, and conveyable. They clearly characterize the particular patterns by which generalized data in an LCS are acquired, represented, fed, interacted, processed, utilized, etc., and thoroughly characterize the macrostructural patterns by which various subsystems of an LCS hold synergic integration in the processes where the LCS autonomously and dynamically performs specified tasks in a hostile environment under spatial, temporal and physical constraints.

It is hard to demonstrate and evaluate a macrostructural modelization problem of LCS by a simple and immediate effort such as numerical computation, simulation or experiment, though the problem may be demonstrated and evaluated by long-term operations of LCS. Nevertheless, besides the unified framework based analysis [43] on typical examples of LCS to visually illustrate the suitability and instantaneousness of the proposed and established block diagram based model, here a case study on the incremental development of the CIMS for a power-station boiler works is presented to demonstrate the potential usability of the proposed and established models of LCS. Macrostructural modelization of the whole CIMS is useful and necessary for overall development planning (including decomposability analysis) and system geometry reduction upon the whole system, and for the communication, coordination, synchronization and integration among subsystem developments.

The specific contributions of this paper can be summarized as follows,

1) The macrostructural modelization problem of LCS has been formulated, highlighting the invariance over the transformation from existing system to target system. This invariance is extraordinarily significant to the development of LCS.

2) Two novel general-purpose information-processing modules, called perception cube and decision spheroid, have been proposed and constructed.

3) A novel block diagram based model has been proposed and established for LCS.

4) For the first time, a level of LCS has been constructively defined through a stratified constructive linkage between a stream of bottom–up growing fusion grammars and a stream of top–down growing synthesis grammars, and accordingly been represented as a context-free grammar.

5) A whole LCS has been represented as a context-free grammar.

6) Comprehensively, a general paradigm has been established and practiced for the investigation into the macrostructural modelization problem of LCS. It is, first building basic elements of characterization, and then structuring these elements toward a whole LCS, furthermore, applying formal language theory to the block diagram based model.

REFERENCES


Huaiglory Tianfield was born in Sichuan, China, in 1966. He received the B.Eng. degree in computer-based industrial automation from Sichuan Institute of Chemical and Light Industry Technology, Sichuan, and the M.Eng. degree in expert systems applications and the Ph.D. degree in intelligent machine systems from East China University of Science and Technology, Shanghai, in 1986, 1989, and 1992, respectively.

Since 2001, he has been Professor of computing, Glasgow Caledonian University, Glasgow, U.K. He was Senior Lecturer in Computing, Gloucester College, Cheltenham, U.K., from 2000 to 2001, Head Researcher of the Team for Distributed Artificial Intelligence, Universität-Essen, Essen, Germany, from 1999 to 2000, Senior Invited Researcher, National Polytechnic University, Grenoble, France, from 1998 to 1999, Research Fellow, Alexander von Humboldt Foundation of Germany, Germany, from 1997 to 1998, and Associate Professor from 1996 to 1998 and Assistant Professor from 1994 to 1995, Tongji University, Shanghai, China, and Postdoctoral Fellow, National Postdoctoral Foundation of China, Shanghai Jiao Tong University, Shanghai, from 1992 to 1994. His research interests include the solution of problems of large complex systems (LCS) by means of the synergy of distributed and artificial intelligence, formalization, software engineering, cybernetics, and systems science. He has published over 80 research papers in refereed journals and international conferences. He has advised many Ph.D. and M.Phil. theses. He served as Chair and an IPC member for many international conferences and symposia and as a Guest Editor for a number of journal special issues.