The Theoretical Framework of Cognitive Informatics

1 author:

Yingxu Wang
The University of Calgary

553 PUBLICATIONS **6,687** CITATIONS

Some of the authors of this publication are also working on these related projects:

Project   Deep Machine Learning and Cognitive Knowledge Learning View project

Project   Cognitive Linguistics: Theories of Formal Semantics and Machine Knowledge Learning View project

The Theoretical Framework of Cognitive Informatics

Yingxu Wang, University of Calgary, Canada

ABSTRACT

Cognitive Informatics (CI) is a transdisciplinary enquiry of the internal information processing mechanisms and processes of the brain and natural intelligence shared by almost all science and engineering disciplines. This article presents an intensive review of the new field of CI. The structure of the theoretical framework of CI is described encompassing the Layered Reference Model of the Brain (LRMB), the OAR model of information representation, Natural Intelligence (NI) vs. Artificial Intelligence (AI), Autonomic Computing (AC) vs. imperative computing, CI laws of software, the mechanism of human perception processes, the cognitive processes of formal inferences, and the formal knowledge system. Three types of new structures of mathematics, Concept Algebra (CA), Real-Time Process Algebra (RTPA), and System Algebra (SA), are created to enable rigorous treatment of cognitive processes of the brain as well as knowledge representation and manipulation in a formal and coherent framework. A wide range of applications of CI in cognitive psychology, computing, knowledge engineering, and software engineering has been identified and discussed.

Keywords: cognitive informatics; theoretical framework; descriptive mathematics; concept algebra; process algebra; system algebra; mathematical models; computing; engineering applications; knowledge engineering; software engineering

INTRODUCTION

The development of classical and contemporary informatics, the cross fertilization between computer science, systems science, cybernetics, computer/software engineering, cognitive science, knowledge engineering, and neuropsychology, has led to an entire range of an extremely interesting and new research field known as Cognitive Informatics (Wang, 2002a, 2003a, b, 2006b; Wang, Johnston & Smith 2002; Wang & Kinsner, 2006). *Informatics* is the science of information that studies the nature of information; it's processing, and ways of transformation between information, matter, and energy.

Definition 1. *Cognitive Informatics* (CI) is a transdisciplinary enquiry of cognitive and information sciences that investigates the

Figure 1. The theoretical framework of CI
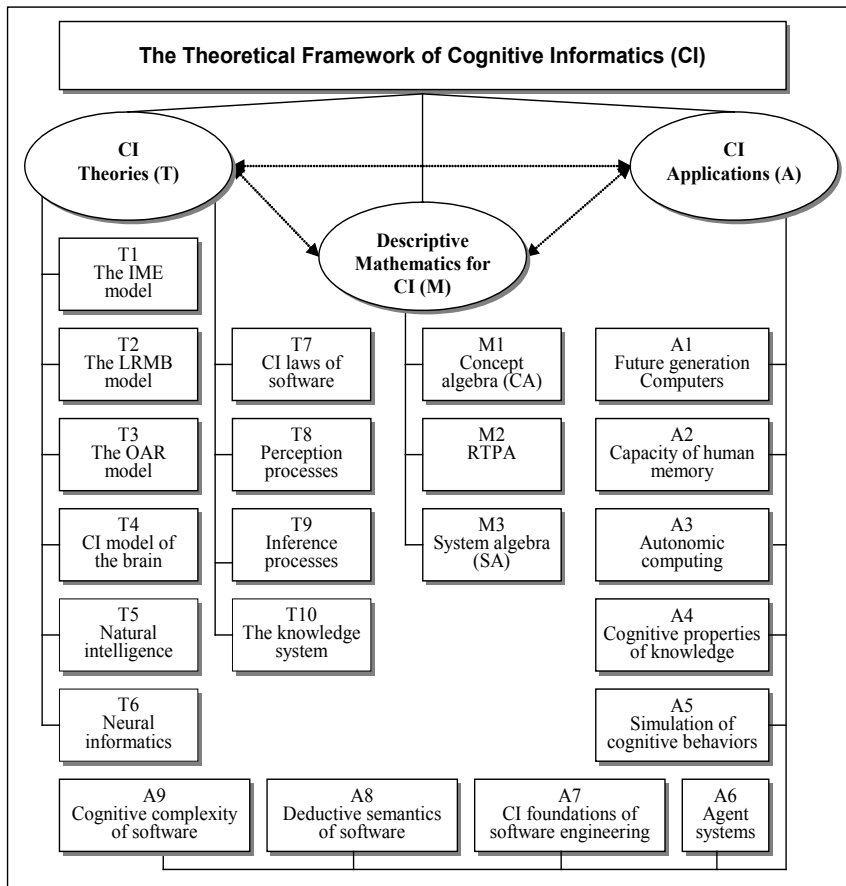
internal information processing mechanisms and processes of the brain and natural intelligence, and their engineering applications via an interdisciplinary approach.

In many disciplines of human knowledge, almost all of the hard problems yet to be solved share a common root in the understanding of the mechanisms of natural intelligence and the cognitive processes of the brain. Therefore, CI is a discipline that forges links between a number of natural science and life science disciplines with informatics and computing science.

The structure of the theoretical framework of CI is described in Figure 1, which covers the Information-Matter-Energy (IME) model (Wang, 2003b), the Layered Reference Model of the Brain (LRMB) (Wang, Wang, Patel & Patel, 2006), the Object-Attribute-Relation (OAR) model of information representation in the brain (Wang, 2006h; Wang & Wang, 2006), the cognitive informatics model of the brain (Wang, Liu, & Wang, 2003; Wang & Wang, 2006), Natural Intelligence (NI) (Wang, 2003b), Autonomic Computing (AC) (Wang, 2004), Neural Informatics (NeI) (Wang, 2002a,

2003b, 2006b), CI laws of software (Wang, 2006f), the mechanisms of human perception processes (Wang, 2005a), the cognitive processes of formal inferences (Wang, 2005c), and the formal knowledge system (Wang, 2006g).

In this article, the theoretical framework of CI is explained in the fundamental theories of CI section. Three structures of new descriptive mathematics such as Concept Algebra (CA), Real-Time Process Algebra (RTPA), and System Algebra (SA) are introduced in the denotational mathematics for CI in order to rigorously deal with knowledge and cognitive information representation and manipulation in a formal and coherent framework. Applications of CI are discussed, which covers cognitive computing, knowledge engineering, and software engineering. Then, it draws conclusions on the theories of CI, the contemporary mathematics for CI, and their applications.

THE FUNDAMENTAL THEORIES OF CI

The fundamental theories of CI encompass 10 transdisciplinary areas and fundamental models, T1 through T10, as identified in Figure 1. This section presents an intensive review of the theories developed in CI, which form a foundation for exploring the natural intelligence and their applications in brain science, neural informatics, computing, knowledge engineering, and software engineering.

The Information-Matter-Energy Model

Information is recognized as the third essence of the natural world supplementing to matter and energy (Wang, 2003b), because the primary function of the human brain is information processing.

Theorem 1. A generic worldview, the IME model states that the natural world (NW) that forms the context of human beings is a dual world: one aspect of it is the physical or the concrete world (PW), and the other is the abstract or the perceptive world (AW), where

matter (M) and energy (E) are used to model the former, and information (I) to the latter, that is:

$$NW \triangleq PW \parallel AW$$
$$= \wp (M, E) \parallel \alpha (I) \qquad (1)$$
$$= \mathfrak{n} (I, M, E)$$

where $\parallel$ denotes a parallel relation, and $\wp$, α, and $\mathfrak{n}$ are functions that determine a certain PW, AW, or NW, respectively, as illustrated in Figure 2.

According to the IME model, information plays a vital role in connecting the physical world with the abstract world. Models of the natural world have been well studied in physics and other natural sciences. However, the modeling of the abstract world is still a fundamental issue yet to be explored in cognitive informatics, computing, software science, cognitive science, brain sciences, and knowledge engineering. Especially the relationships between I-M-E and their transformations are deemed as one of the fundamental questions in CI.

Corollary 1. The natural world $NW(I, M, E)$, particularly part of the abstract world, $AW(I)$, is cognized and perceived differently by individuals because of the uniqueness of perceptions and mental contexts among people.

Corollary 1 indicates that although the physical world $PW(M, E)$ is the same to everybody, the natural world $NW(I, M, E)$ is unique to different individuals because the abstract world $AW(I)$, as a part of it, is subjective depending on the information an individual obtains and perceives.

Corollary 2. The *principle of transformability between IME* states that, according to the IME model, the three essences of the world are predicated to be transformable between each other as described by the following generic functions f_1 to f_6:

$$I = f_1(M) \qquad (2.1)$$
$$M = f_2(I) \square f_1^{-1}(I) \qquad (2.2)$$

Figure 2. The IME model of the worldview
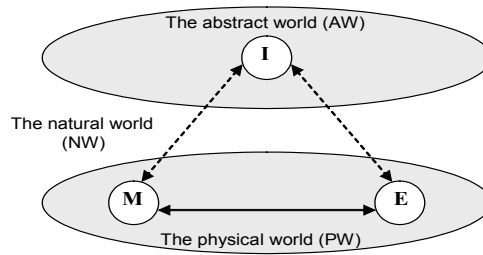
$$I = f_3(E) \tag{2.3}$$
$$E = f_4(I) \,\square\, f_3^{-1}(I) \tag{2.4}$$
$$E = f_5(M) \tag{2.5}$$
$$M = f_6(E) = f_5^{-1}(E) \tag{2.6}$$

where a question mark on the equal sign denotes an uncertainty if there exists such a reverse function (Wang, 2003b).

Albert Einstein revealed Functions f_5 and f_6, the relationship between matter (m) and energy (E), in the form $E = mC^2$, where C is the speed of light. It is a great curiosity to explore what the remaining relationships and forms of transformation between I-M-E will be. To a certain extent, cognitive informatics is the science to seek possible solutions for f_1 to f_4. A clue to explore the relations and transformability is believed in the understanding of the natural intelligence and its information processing mechanisms in CI.

Definition 2. *Information* in CI is defined as a generic abstract model of properties or attributes of the natural world that can be distinctly elicited, generally abstracted, quantitatively represented, and mentally processed.

Definition 3. The *measurement of information*, I_k, is defined by the cost of code to abstractly represent a given size of internal message X in the brain in a digital system based on k, that is:

$$I_k = f : X \rightarrow S_k \tag{3}$$
$$= \lceil \log_k X \rceil$$

where I_k is the content of information in a k-based digital system, and S_k is the measurement scale based on k. The unit of I_k is the number of k-based digits (Wang, 2003b).

Equation 3 is a generic measure of information sizes. When a binary digital representation system is adopted, that is $k = b = 2$, it becomes the most practical one as follows.

Definition 4. The metalevel representation of information, I_b, is that when $k = b = 2$, that is:

$$I_b = f : X \rightarrow S_b \tag{4}$$
$$= \lceil \log_b X \rceil$$

where the unit of information, I_b, is a *bit*.

Note that the *bit* here is a concrete and deterministic unit, and it is no longer probability-based as in conventional information theories (Bell, 1953; Shannon, 1948). To a certain extent, computer science and engineering is a branch of modern informatics that studies machine representation and processing of external information; while CI is a branch of contemporary informatics that studies internal information representation and processing in

Figure 3. LRMB model
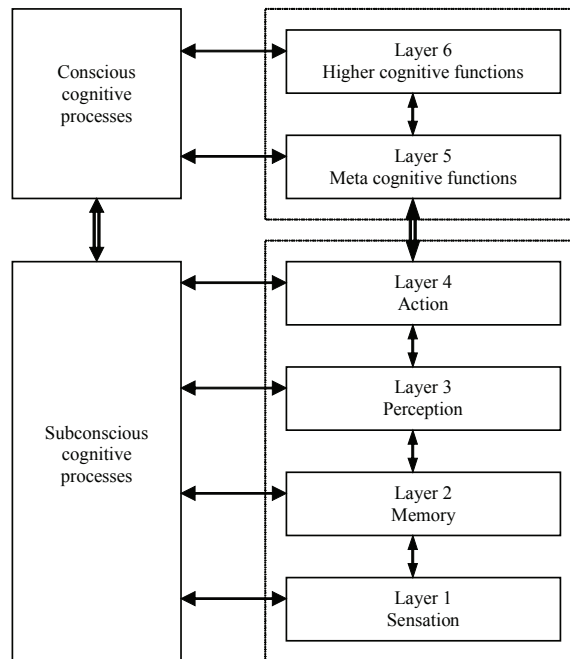
the brain.

Theorem 2. The most fundamental form of information that can be represented and processed is binary digit where $k = b = 2$.

Theorem 2 indicates that any form of information in the physical (natural) and abstract (mental) worlds can be unified on the basis of binary data. This is the CI foundation of modern digital computers and NI.

The Layered Reference Model of the Brain

The LRMB (Wang et al., 2006) is developed to explain the fundamental cognitive mechanisms and processes of natural intelligence. Because a variety of life functions and cognitive processes have been identified in

CI, psychology, cognitive science, brain science, and neurophilosophy, there is a need to organize all the recurrent cognitive processes in an integrated and coherent framework. The LRMB model explains the functional mechanisms and cognitive processes of natural intelligence that encompasses 37 cognitive processes at six layers known as the *sensation, memory, perception, action, metacognitive,* and *higher cognitive layers* from the bottom-up as shown in Figure 3. LRMB elicits the core and highly repetitive recurrent cognitive processes from a huge variety of life functions, which may shed light on the study of the fundamental mechanisms and interactions of complicated mental processes, particularly the relationships and interactions between the inherited and the acquired life functions as well as those of the subconscious and conscious cognitive

processes.

The OAR Model of Information Representation in the Brain

Investigation into the cognitive models of information and knowledge representation in the brain is perceived to be one of the fundamental research areas that help to unveil the mechanisms of the brain. The *Object-Attribute-Relation* (OAR) model (Wang, 2006h; Wang et al., 2003) describes human memory, particularly the long-term memory, by using the *relational metaphor*, rather than the traditional *container metaphor* that used to be adopted in psychology, computing, and information science. The OAR model shows that human memory and knowledge are represented by relations, that is, connections of synapses between neurons, rather than by the neurons themselves as the traditional container metaphor described. The OAR model can be used to explain a wide range of human information processing mechanisms and cognitive processes.

The Cognitive Informatics Model of the Brain

The human brain and its information processing mechanisms are centred in CI. A cognitive informatics model of the brain is proposed in Wang and Wang (2006), which explains the natural intelligence via interactions between the inherent (subconscious) and acquired (conscious) life functions. The model demonstrates that memory is the foundation for any natural intelligence. Formalism in forms of mathematics, logic, and rigorous treatment is introduced into the study of cognitive and neural psychology and natural informatics. Fundamental cognitive mechanisms of the brain, such as the architecture of the thinking engine, internal knowledge representation, long-term memory establishment, and roles of sleep in long-term memory development have been investigated (Wang & Wang, 2006).

Natural Intelligence (NI)

Natural Intelligence (NI) is the domain of CI. Software and computer systems are recognized as a subset of intelligent behaviors of human beings described by programmed instructive information (Wang, 2003b; Wang & Kinsner, 2006). The relationship between Artificial Intelligence (AI) and NI can be described by the following theorem.

Theorem 3. The law of *compatible intelligent capability* states that *artificial intelligence (AI)* is always a subset of the *natural intelligence* (NI), that is:

$$AI \subseteq NI \qquad (5)$$

Theorem 3 indicates that AI is dominated by NI. Therefore, one should not expect a computer or a software system to solve a problem where humans cannot. In other words, no AI or computing system may be designed and/or implemented for a given problem where there is no solution being known by human beings.

Neural Informatics (NeI)

Definition 5. *Neural Informatics* (NeI) is a new interdisciplinary enquiry of the biological and physiological representation of information and knowledge in the brain at the neuron level and their abstract mathematical models (Wang, 2004; Wang & Wang, 2006).

NeI is a branch of CI, where memory is recognized as the foundation and platform of any natural or artificial intelligence (Wang & Wang, 2006).

Definition 6. The *Cognitive Models of Memory* (CMM) states that the architecture of human memory is parallel configured by the Sensory Buffer Memory (SBM), Short-Term Memory (STM), Long-Term Memory (LTM), and Action-Buffer Memory (ABM), that is:

$$CMM \triangleq SBM \\ \| STM \\ \| LTM \\ \| ABM \qquad (6)$$

where the ABM is newly identified in Wang and Wang (2006).

The major organ that accommodates memories in the brain is the cerebrum or the cerebral cortex. In particular, the association and premotor cortex in the frontal lobe, the temporal lobe, sensory cortex in the frontal lobe, visual cortex in the occipital lobe, primary motor cortex in the frontal lobe, supplementary motor area in the frontal lobe, and procedural memory in cerebellum (Wang & Wang, 2006).

The CMM model and the mapping of the four types of human memory onto the physiological organs in the brain reveal a set of fundamental mechanisms of NeI. The OAR model of information/knowledge representation described in the OAR model of information representation in the brain section provides a generic description of information/knowledge representation in the brain (Wang, 2006h; Wang et al., 2003).

The theories of CI and NeI explain a number of important questions in the study of NI. Enlightening conclusions derived in CI and NeI are such as: (a) LTM establishment is a subconscious process; (b) The long-term memory is established during sleeping; (c) The major mechanism for LTM establishment is by sleeping; (d) The general acquisition cycle of LTM is equal to or longer than 24 hours; (e) The mechanism of LTM establishment is to update the entire memory of information represented as an OAR model in the brain; and (f) Eye movement and dreams play an important role in LTM creation. The latest development in CI and NeI has led to the determination of the magnificent and expected capacity of human memory as described in the Estimation of the Capacity of Human Memory section.

Cognitive Informatics Laws of Software

It is commonly conceived that software as an artifact of human creativity is not constrained by the laws and principles discovered in the physical world. However, it is unknown what constrains software. The new informatics metaphor proposed by the author in CI perceives software is a type of instructive and behavioral information. Based on this, it is asserted that software obeys the laws of informatics. A comprehensive set of 19 CI laws for software have been established in Wang (2006f), such as:

1. Abstraction
2. Generality
3. Cumulativeness
4. Dependency on cognition
5. Three-dimensional behavior space known as the object (O), space (S), and time (T)
6. Sharability
7. Dimensionless
8. Weightless
9. Transformability between I-M-E
10. Multiple representation forms
11. Multiple carrying media
12. Multiple transmission forms
13. Dependency on media
14. Dependency on energy
15. Wearless and time dependency
16. Conservation of entropy
17. Quality attributes of informatics
18. Susceptible to distortion
19. Scarcity

The informatics laws of software extend the knowledge on the fundamental laws and properties of software where the conventional product metaphor could not explain. Therefore, CI forms one of the foundations of software engineering and computing science.

Mechanisms of Human Perception Processes

Definition 7. *Perception* is a set of interpretive cognitive processes of the brain at the subconscious cognitive function layers that detects, relates, interprets, and searches internal cognitive information in the mind.

Perception may be considered as the *sixth sense* of human beings, which almost all cognitive life functions rely on. Perception is also an important cognitive function at the subconscious layers that determines personal-

ity. In other words, personality is a faculty of all subconscious life functions and experience cumulated via conscious life functions.

According to LRMB, the main cognitive processes at the perception layer are emotion, motivation, and attitude (Wang, 2005a). The relationship between the internal emotion, motivation, attitude, and the embodied external behaviors can be formally and quantitatively described by the *motivation/attitude-driven behavioral* (MADB) *model* (Wang & Wang, 2006), which demonstrates that complicated psychological and cognitive mental processes may be formally modeled and rigorously described by mathematical means (Wang, 2002b, 2003d, 2005c).

The Cognitive Processes of Formal Inferences

Theoretical research is predominately an inductive process, while *applied research* is mainly a deductive one. Both inference processes are based on the cognitive process and means of abstraction. *Abstraction* is a powerful means of philosophy and mathematics. It is also a preeminent trait of the human brain identified in CI studies (Wang, 2005c). All formal logical inferences and reasonings can only be carried out on the basis of abstract properties shared by a given set of objects under study.

Definition 8. *Abstraction* is a process to elicit a subset of objects that shares a common property from a given set of objects and to use the property to identify and distinguish the subset from the whole in order to facilitate reasoning.

Abstraction is a gifted capability of human beings. Abstraction is a basic cognitive process of the brain at the metacognitive layer according to LRMB (Wang et al., 2006). Only by abstraction can important theorems and laws about the objects under study be elicited and discovered from a great variety of phenomena and empirical observations in an area of inquiry.

Definition 9. *Inferences* are a formal cognitive process that reasons a possible causality from given premises based on known causal relations between a pair of cause and effect proven true by empirical arguments, theoretical inferences, or statistical regulations.

Formal inferences may be classified into the deductive, inductive, abductive, and analogical categories (Wang, 2005c). *Deduction* is a cognitive process by which a specific conclusion necessarily follows from a set of general premises. *Induction* is a cognitive process by which a general conclusion is drawn from a set of specific premises based on three designated samples in reasoning or experimental evidences. *Abduction* is a cognitive process by which an inference to the best explanation or most likely reason of an observation or event. *Analogy* is a cognitive process by which an inference about the similarity of the same relations holds between different domains or systems, and/or examines that if two things agree in certain respects, then they probably agree in others. A summary of the formal definitions of the five inference techniques is shown in Table 1.

For seeking generality and universal truth, either the objects or the relations can only be abstractly described and rigorously inferred by abstract models rather than real-world details.

The Formal Knowledge System

Mathematical thoughts (Jordan & Smith, 1997) provide a successful paradigm to organize and validate human knowledge, where once a truth or a theorem is established, it is true until the axioms or conditions that it stands for are changed or extended. A proven truth or theorem in mathematics does not need to be argued each time one uses it. This is the advantage and efficiency of formal knowledge in science and engineering. In other words, if any theory or conclusion may be argued from time-to-time based on a wiser idea or a trade-off, it is an empirical result rather than a formal one.

The Framework of Formal Knowledge (FFK) of mankind (Wang, 2006g) can be

Table 1. Definitions of formal inferences

No.	Inference technique	Formal description		Usage
		Primitive form	**Composite form**	
1	Abstraction	$\forall S,\ p \Rightarrow \exists\ e \in E \subseteq S,\ p(e)$	-	To elicit a subset of elements with a given generic property.
2	Deduction	$\forall x \in \mathbf{X},\ p(x) \Rightarrow \exists a \in \mathbf{X}, p(a)$	$(\forall x \in \mathbf{X}, p(x) \Rightarrow q(x))\ \forall x \in \mathbf{X}, p(x) \Rightarrow \exists a \in \mathbf{X}, p(a)\ (\exists a \in \mathbf{X}, p(a) \Rightarrow q(a))$	To derive a conclusion based on a known and generic premises.
3	Induction	$((\exists a \in \mathbf{X}, P(a)) \wedge (\exists k, k{+}1 \in \mathbf{X}, (P(k) \Rightarrow P(k{+}1)))) \Rightarrow \forall x \in \mathbf{X}, P(x)$	$((\exists a \in \mathbf{X},\ p(a) \Rightarrow q(a)) \wedge (\exists k,\ k{+}1 \in \mathbf{X},\ ((p(k) \Rightarrow q(k)) \Rightarrow (p(k{+}1) \Rightarrow q(k{+}1)))) \Rightarrow \forall x \in \mathbf{X}, p(x) \Rightarrow q(x)$	To determine the generic behavior of the given list or sequence of recurring patterns by three samples.
4	Abduction	$(\forall x \in \mathbf{X}, p(x) \Rightarrow q(x)) \Rightarrow (\exists a \in \mathbf{X}, q(a) \Rightarrow p(a))$	$(\forall x \in \mathbf{X}, p(x) \Rightarrow q(x) \wedge r(x) \Rightarrow q(x)) \Rightarrow (\exists a \in \mathbf{X}, q(a) \Rightarrow (p(a) \vee r(a)))$	To seek the most likely cause(s) and reason(s) of an observed phenomenon.
5	Analogy	$\exists a \in \mathbf{X},\ p(a) \Rightarrow \exists b \in \mathbf{X}, p(b)$	$(\exists a \in \mathbf{X}, p(a) \Rightarrow q(a)) \Rightarrow (\exists b \in \mathbf{X}, p(b) \Rightarrow q(b))$	To predict a similar phenomenon or consequence based on a known observation.

described as shown in Figure 5. An FFK is centered by a set of theories. A *theory* is a statement of how and why certain objects, facts, or truths are related. All objects in nature and their relations are constrained by invariable laws, no matter if one observed them or not at any given time. An *empirical truth* is a truth based on or verifiable by observation, experiment, or experience. A *theoretical proposition* is an assertion based on formal theories or logical reasoning. Theoretical knowledge is a formalization of generic truth and proven abstracted empirical knowledge. Theoretical knowledge may be easier to acquire when it exists. However, empirical knowledge is very difficult to be gained without hands-on practice.

According to the FFK model, an immature discipline of science and engineering is characterized by its body of knowledge not being formalized. Instead of a set of proven theories, the immature disciplines document a large set of observed facts, phenomena, and their possible or partially working explanations and hypotheses. In such disciplines, researchers and practitioners might be able to argue every informal conclusion documented in natural languages from time-to-time probably for hundreds of years, until it is formally described in mathematical forms and proved rigorously.

The disciplines of mathematics and physics are successful paradigms that adopt the FFK formal knowledge system. The key advantages of the formal knowledge system are its stability and efficiency. The former is a property of the formal knowledge that once it is established and formally proved, users who refers to it will no longer need to reexamine or reprove it. The latter is a property of formal knowledge that is exclusively true or false that saves everybody's time from arguing a proven theory.

DENOTATIONAL MATHEMATICS FOR CI

The history of sciences and engineering shows that new problems require new forms of mathematics. CI is a new discipline, and the problems in it require new mathematical means that are descriptive and precise in expressing and denoting human and system actions and

Figure 4. The framework of formal knowledge (FFK)
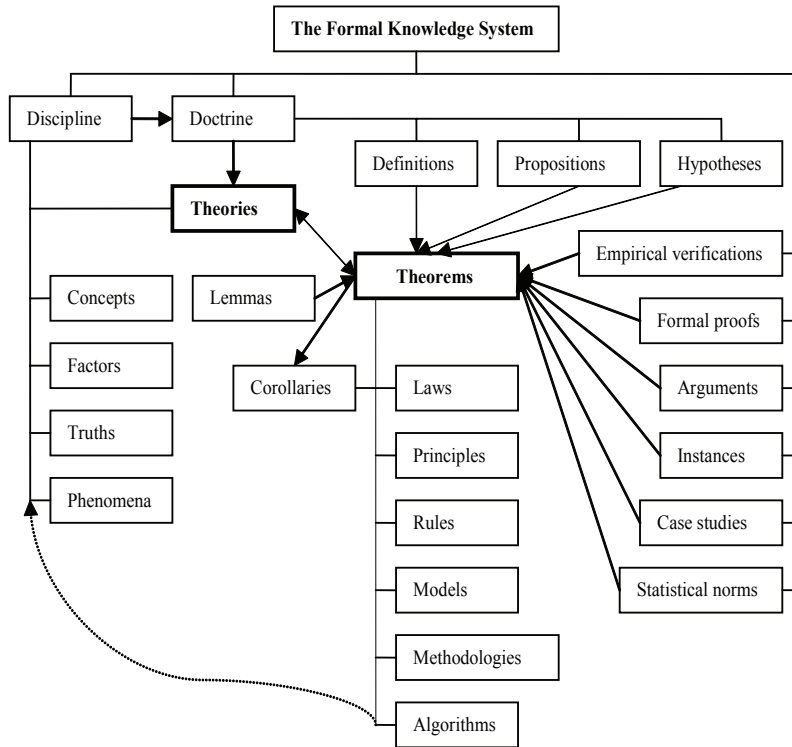
behaviors. Conventional *analytic mathematics* are unable to solve the fundamental problems inherited in CI and related disciplines such as neuroscience, psychology, philosophy, computing, software engineering, and knowledge engineering. Therefore, *denotational mathematical structures and means* (Wang, 2006c) beyond mathematical logic are yet to be sought.

Although there are various ways to express facts, objects, notions, relations, actions, and behaviors in natural languages, it is found in CI that human and system behaviors may be classified into three basic categories known as to *be*, to *have*, and to *do*. All mathematical means and forms, in general, are an abstract and formal description of these three categories of expressibility and their rules. Taking this view, mathematical logic may be perceived as the ab-

stract means for describing "to be," set theory describing "to have," and algebras, particularly process algebra, describing "to do."

Theorem 4. The utility of mathematics is the means and rules to express thought rigorously and generically at a higher level of abstraction.

Three types of new mathematics, Concept Algebra (CA), Real-Time Process Algebra (RTPA), and System Algebra (SA), are created in CI to enable rigorous treatment of knowledge representation and manipulation in a formal and coherent framework. The three new structures of contemporary mathematics have extended the abstract objects under study in mathematics from basic mathematical entities of numbers

Figure 5. The structural model of an abstract concept
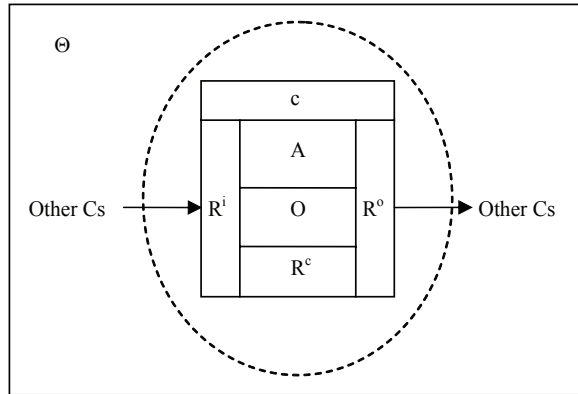
and sets to a higher level, that is, concepts, behavioral processes, and systems. A wide range of applications of the denotational mathematics in the context of CI has been identified (Wang, 2002b, 2006d, e).

Concept Algebra

A *concept* is a cognitive unit (Ganter & Wille, 1999; Quillian, 1968; Wang, 2006e) by which the meanings and semantics of a real-world or an abstract entity may be represented and embodied based on the OAR model.

Definition 10. An *abstract concept c* is a 5-tuple, that is:

$$c \triangleq (O, A, R^c, R^i, R^o) \qquad (7)$$

where

- *O* is a nonempty set of object of the concept, $O = \{o_1, o_2, ..., o_m\} = ÞU$, where Þ U denotes a power   set of *U*.
- *A* is a nonempty set of attributes, $A = \{a_1, a_2, ..., a_n\} = ÞM$.
- $R^c \subseteq O \times A$ is a set of internal relations.
- $R^i \subseteq C' \times C$ is a set of input relations, where C′ is a set of external concepts.
- $R^o \subseteq C \times C'$ is a set of output relations.

A structural concept model of $c = (O, A, R^c, R^i, R^o)$ can be illustrated in Figure 6, where *c, A, O,* and *R, R* = $\{R^c, R^i, R^o\}$, denote the concept, its attributes, objects, and internal/external relations, respectively.

Definition 11. *Concept algebra* is a new mathematical structure for the formal treatment of abstract concepts and their algebraic relations, operations, and associative rules for composing complex concepts and knowledge (Wang, 2006e).

Concept algebra deals with the algebraic relations and associational rules of abstract concepts. The associations of concepts form a foundation to denote complicated relations between concepts in knowledge representation. The associations among concepts can be classified into nine categories, such as inheritance, extension, tailoring, substitute, composition, decomposition, aggregation, specification, and instantiation as shown in Figure 6 and Table 2 (Wang, 2006e). In Figure 6, $R = \{R^c, R^i, R^o\}$, and all nine associations describe composing rules among concepts, except instantiation that is a relation between a concept and a specific object.

Definition 12. A *generic knowledge K* is an

Figure 6. The nine concept association operations as knowledge composing rules
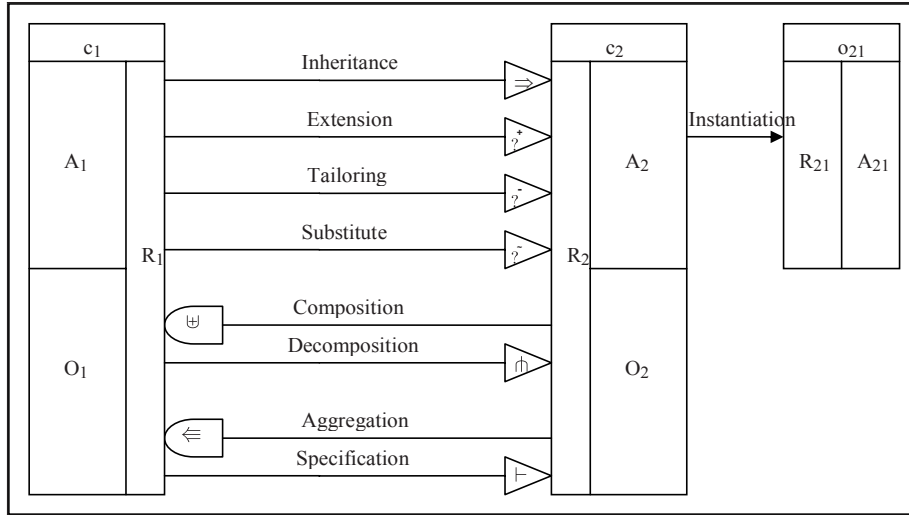
n-nary relation R_k among a set of n multiple concepts in C, that is

$$K = R_k : (\overset{n}{\underset{i=1}{X}} C_i) \rightarrow C \qquad (8)$$

where $\overset{n}{\underset{i=1}{U}} C_i = C$, and
$R_k \in \Re = \{\Rightarrow, \overset{+}{\Rightarrow}, \overline{\Rightarrow}, \overset{\sim}{\overline{\Rightarrow}}, \uplus, \pitchfork, \Leftarrow, \vdash, \mapsto\}$.

In Definition 12, the relation R_k is one of the concept operations in CA as defined in Table 2 (Wang, 2006e) that serves as the knowledge composing rules.

Definition 13. A *concept network CN* is a hierarchical network of concepts interlinked by the set of nine associations $\Re$ defined in CA, that is:

$$CN = R_{\hat{e}} : \overset{n}{\underset{i=1}{X}} C_i \rightarrow \overset{n}{\underset{i=j}{X}} C_j \qquad (9)$$

where $R_k \in \Re$.

Because the relations between concepts are transitive, the generic topology of knowledge is a hierarchical concept network. The advantages of the hierarchical knowledge architecture K in the form of concept networks are as follows: (a) *Dynamic*: The knowledge networks may be updated dynamically along with information acquisition and learning without destructing the existing concept nodes and relational links. (b) *Evolvable*: The knowledge networks may grow adaptively without changing the overall and existing structure of the hierarchical network.

A summary of the algebraic relations and operations of concepts defined in CA are provided in Table 2.

Real-Time Process Algebra (RTPA)

A key metaphor in system modeling, specification, and description is that a software system can be perceived and described as the *composition* of a set of interacting *processes*. Hoare (1985), Milner (1989), and others developed various algebraic approaches to represent communicating and concurrent systems, known as process algebra. A *process algebra* is a set of

formal notations and rules for describing algebraic relations of software processes. *Real-Time Process Algebra* (Wang, 2002b, 2005b) extends process algebra to time/event, architecture, and system dispatching manipulations in order to formally describe and specify architectures and behaviors of software systems. A process in RTPA is a computational operation that transforms a system from a state to another by changing its inputs, outputs, and/or internal variables. A process can be a single metaprocess or a complex process formed by using the process combination rules of RTPA known as process relations.

Definition 14. *Real-Time Process Algebra* is a set of formal notations and rules for describing algebraic and real-time relations of software processes.

RTPA models 17 metaprocesses and 17 process relations. A metaprocess is an elementary and primary process that serves as a common and basic building block for a software system. Complex processes can be derived from metaprocesses by a set of process relations that serves as process combinatory rules. Detailed semantics of RTPA may be referred to in Wang (2002b).

Program modeling is on coordination of computational behaviors with given data objects. Behavioral or instructive knowledge can be modeled by RTPA. A generic program model can be described by a formal treatment of statements, processes, and complex processes from the bottom-up in the program hierarchy.

Definition 15. A *process P* is a composed listing and a logical combination of *n* metastatements p_i and p_j, $1 \le i < n, 1 < j \le m = n+1$, according to certain composing relations r_{ij}, that is:

$$P = \overset{n-1}{\underset{i=1}{R}}(p_i \; r_{ij} \; p_j), j = i+1$$
$$= (...(((p_1) \; r_{12} \; p_2) \; r_{23} \; p_3) \; ... \; r_{n-1,n} \; p_n) \tag{10}$$

where the big-R notation (Wang, 2002b, 2006i) is adopted to describes the nature of processes as the building blocks of programs.

Definition 16. A *program* $\mathfrak{P}$ is a composition of a finite set of *m* processes according to the time-, event-, and interrupt-based process dispatching rules, that is:

$$\mathfrak{P} = \overset{m}{\underset{k=1}{R}}(@ \, e_k \hookmapsto P_k) \tag{11}$$

Equations 9.1 and 10.1 indicate that a program is an *embedded relational algebraic* entity. A statement *p* in a program is an instantiation of a metainstruction of a programming language that executes a basic unit of coherent function and leads to a predictable behavior.

Theorem 5. The *embedded relational model (ERM)* states that a software system or a program $\mathfrak{P}$ is a set of complex embedded relational processes, in which all previous processes of a given process form the context of the current process, that is:

$$\mathfrak{P} = \overset{m}{\underset{k=1}{R}}(@ \, e_k \hookmapsto P_k)$$
$$= \overset{m}{\underset{k=1}{R}}[@ \, e_k \hookmapsto \overset{n-1}{\underset{i=1}{R}}(p_i(k) \; r_{ij}(k) \; p_j(k))], j = i+1 \tag{12}$$

ERM presented in Theorem 5 provides a *unified mathematical model of programs* (Wang, 2006a) for the first time, which reveals that a program is a finite and nonempty set of embedded binary relations between a current statement and *all previous ones* that formed the *semantic context* or environment of computing.

Definition 17. A *metaprocess* is the most basic and elementary processes in computing that cannot be broken up further. The set of *metaprocesses P* encompasses 17 fundamental primitive operations in computing as follows:

$$P = \{:=, \blacklozenge, \Rightarrow, \Leftarrow, \Leftarrow, >, <, |>, |<, \underline{@}, \triangleq, \uparrow,$$
$$\downarrow, !, \oslash, \square, \S\}$$

(13)

Definition 18. A *process relation* is a composing rule for constructing complex processes by using the metaprocesses. The *process relations* **R** of RTPA are a set of 17 composing operations and rules to built larger architectural components and complex system behaviors using the metaprocesses, that is:

$$R = \{\rightarrow, \curvearrowright, |, |...|, R^*, R^+, R^i, \circlearrowright, \rightarrowtail, \|,$$
$$\square, \|\|, », \lightning, \hookrightarrow_p, \hookrightarrow_e, \hookrightarrow_i\}$$

(14)

The definitions, syntaxes, and formal semantics of each of the metaprocesses and process relations may be referred to RTPA (Wang, 2002b, 2006f). A complex process and a program can be derived from the metaprocesses by the set of algebraic process relations. Therefore, a program is a set of embedded relational processes as described in Theorem 5.

A summary of the metaprocesses and their algebraic operations in RTPA are provided in Table 2.

System Algebra (SA)

Systems are the most complicated entities and phenomena in the physical, information, and social worlds across all science and engineering disciplines (Klir, 1992; von Bertalanffy, 1952; Wang, 2006d). Systems are needed because the physical and/or cognitive power of an individual component or person is not enough to carry out a work or solving a problem. An *abstract system* is a collection of coherent and interactive entities that has stable functions and clear boundary with external environment. An abstract system forms the generic model of various real world systems and represents the most common characteristics and properties of them.

Definition 19. *System algebra* is a new abstract mathematical structure that provides an algebraic treatment of abstract systems as well as their relations and operational rules for forming complex systems (Wang, 2006d).

Abstract systems can be classified into two categories known as the closed and open systems. Most practical and useful systems in nature are open systems in which there are interactions between the system and its environment. However, for understanding easily, the closed system is introduced first.

Definition 20. A *closed system* $\widehat{S}$ is a 4-tuple, that is :

$$\widehat{S} = (C, R, B, \Omega)$$

(15)

where

- C is a nonempty set of components of the system, $C = \{c_1, c_2, ..., c_n\}$.
- R is a nonempty set of relations between pairs of the components in the system, $R = \{r_1, r_2, ..., r_m\}$, $R \subseteq C \times C$.
- B is a set of behaviors (or functions), $B = \{b_1, b_2, ..., b_p\}$.
- Ω is a set of constraints on the memberships of components, the conditions of relations, and the scopes of behaviors, $\Omega = \{\omega_1, \omega_2, ..., \omega_q\}$.

Most practical systems in the real world are not closed. That is, they need to interact with external world known as the *environment* Θ in order to exchange energy, matter, and/or information. Such systems are called open systems. Typical interactions between an open system and the environment are inputs and outputs.

Definition 21. An *open system* S is a 7-tuple, that is:

$$S = (C, R, B, \Omega, \Theta)$$
$$= (C, R^c, R^i, R^o, B, \Omega, \Theta)$$

(16)

where the extensions of entities beyond the closed system are as follows:

Figure 7. The abstract model of an open system
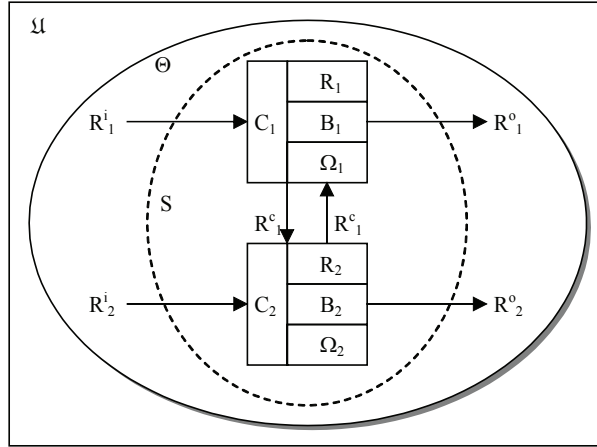
- Θ is the environment of S with a nonempty set of components C_Θ outside C.
- $R^c \subseteq C \times C$ is a set of internal relations.
- $R^i \subseteq C_\Theta \times C$ is a set of external input relations.
- $R^o \subseteq C \times C_\Theta$ is a set of external output relations.

An open system $S = (C, R^c, R^i, R^o, B, \Omega, \Theta)$ can be illustrated in Figure 7 (Wang, 2006d).

Theorem 6. The equivalence between open and closed systems states that an open system S is equivalent to a closed system $\hat{S}$, or vice verse, when its environment Θ_S or $\Theta_{\hat{S}}$ is conjoined, respectively, that is:

$$\begin{cases} \hat{S} = S \sqcup \Theta_S \\ S = \hat{S} \sqcup \Theta_{\hat{S}} \end{cases} \qquad (17)$$

According to Theorem 6, any subsystem $\widehat{S_k}$ of a closed system $\hat{S}$ is an open system S. That is, any supersystem S of a given set of n open systems S_k, plus their environments Θ_k,

$1 \le k \le n$, is a closed system. The algebraic relations and operations of systems in SA are summarized in Table 2.

Theorem 7. The Wang's *first law* of system science, *system fusion*, states that system conjunction or composition between two systems S_1 and S_2 creates *new relations* ΔR_{12} and/or *new behaviors* (functions) ΔB_{12} that are solely a property of the new supersystem S determined by the sizes of the two intersected component sets $\#(C_1)$ and $\#(C_2)$, that is:

$$\begin{aligned} \Delta R_{12} &= \#(R) - (\#(R_1) + \#(R_2)) \\ &= (\#(C_1 + C_2))^2 - ((\#(C_1))^2 + (\#(C_2))^2) \\ &= 2\,(\#(C_1) \bullet \#(C_2)) \end{aligned}$$

$$(18)$$

The discovery in Theorem 7 reveals that the mathematical explanation of system utilities is the newly gained relations ΔR_{12} and/or behaviors (functions) ΔB_{12} during the conjunction of two systems or subsystems. The empirical awareness of this key system property has been intuitively or qualitatively observed for centuries. However, Theorem 7 is the first rigorous explanation of the mechanism of system gains during system conjunctions and

compositions. According to Theorem 7, the maximum *incremental* or *system gain* equals to the number of by-directly interconnection between all components in both S_1 and S_2, that is, $2(\#(C_1) \bullet \#(C_2))$.

Theorem 8. The Wang's *2nd law* of system science, the *maximum system gain*, states that work done by a system is always larger than any of its components, but is less than or is equal to the sum of those of its components, that is:

$$\begin{cases} W(S) \leq \sum_{i=1}^{n} W(e_i), & \text{h} \leq 1 \\ W(S) > \max(W(e_i)), & e_i \in E_S \end{cases} \quad (19)$$

There was a myth on an ideal system in conventional systems theory that supposes the work down by the ideal system $W(S)$ may be greater than the sum of all its components

$W(e_i)$, that is: $W(S) \geq \sum_{i=1}^{n} W(e_i)$. According to Theorems 7 and 8, the ideal system utility is impossible to achieve.

A summary of the algebraic operations and their notations in CA, RTPA, and SA is provided in Table 2. Details may be referred to in Wang (2006d, g).

APPLICATIONS OF CI

The last two sections have reviewed the latest development of fundamental researches in CI, particularly its theoretical framework and descriptive mathematics. A wide range of applications of CI has been identified in multidisciplinary and transdisciplinary areas, such as: (1) The architecture of future generation computers; (2) Estimation the capacity of human memory; (3) Autonomic computing; (4) Cognitive properties of information, data, knowledge, and skills in knowledge engineering; (5) Simulation of human cognitive behaviors using descriptive mathematics; (6) Agent systems; (7) CI foundations of software engi-

neering; (8) Deductive semantics of software; and (9) Cognitive complexity of software.

The Architecture of Future Generation Computers

Conventional machines are invented to extend human physical capability, while modern information processing machines, such as computers, communication networks, and robots, are developed for extending human intelligence, memory, and the capacity for information processing (Wang, 2004). Recent advances in CI provide formal description of an entire set of cognitive processes of the brain (Wang et al., 2006). The fundamental research in CI also creates an enriched set of contemporary *denotational mathematics* (Wang, 2006c), for dealing with the extremely complicated objects and problems in natural intelligence, neural informatics, and knowledge manipulation.

The theory and philosophy behind the next generation computers and computing methodologies are CI (Wang, 2003b, 2004). It is commonly believed that the future-generation computers, known as the cognitive computers, will adopt non-von Neumann (von Neumann, 1946) architectures. The key requirements for implementing a conventional *stored-program controlled* computer are the generalization of common computing architectures and the computer is able to interpret the data loaded in memory as computing instructions. These are the essences of stored-program controlled computers known as the von Neumann (1946) architecture. Von Neumann elicited five fundamental and essential components to implement general-purpose programmable digital computers in order to embody the concept of stored-program-controlled computers.

Definition 22. A *von Neumann Architecture* (VNA) of computers is a 5-tuple that consists of the components: (a) the *arithmetic-logic unit* (ALU), (b) the *control unit* (CU) with a *program counter* (PC), (c) a *memory* (M), (d) a set of *input/output* (*I/O*) *devices*, and (e) a *bus* (B) that provides the data path between these components, that is:

Table 2. Taxonomy of contemporary mathematics for knowledge representation and manipulation

Operations	Concept Algebra	System Algebra	Real-Time Process Algebra			
			Meta Processes		Relational Operations	
Super/subrelation	≻ / ≺	⊒ / ⊑	Assignment	:=	Sequence	→
Related/independent	↔ / ↮	↔ / ↮	Evaluation	◆	Jump	↷
Equivalent	=	=	Addressing	⇒	Branch	⌐
Consistent	≅		Memory allocation	⇐	Switch	| … | …
Overlapped		Π	Memory release	⇍	While-loop	R^{*}
Conjunction	+	⊔	Read	≫	Repeat-loop	R^{+}
Elicitation	*		Write	≪	For-loop	R^{i}
Comparison	~		Input	|≫	Recursion	↻
Definition	≜		Output	|≪	Procedure call	↣
Difference		⊟	Timing	@	Parallel	‖
Inheritance	⇒	⇒	Duration	≜	Concurrence	□
Extension	⇒+	⇒+	Increase	↑	Interleave	⫴
Tailoring	⇛	⇛	Decrease	↓	Pipeline	»
Substitute	⇒~	⇒~	Exception detection	!	Interrupt	⚡
Composition	⊎	⊎	Skip	⊘	Time-driven dispatch	↪ₜ
Decomposition	⅄	⅄	Stop	□	Event-driven dispatch	↪ₑ
Aggregation/ generalization	⇐	⇐	System	§	Interrupt-driven dispatch	↪ᵢ
Specification	⊢	⊢				
Instantiation	↦	↦				

$$VNA \triangleq (ALU, CU, M, I/O, B) \qquad (20)$$

Definition 23. *Conventional computers* with VNA are aimed at stored-program-controlled *data* processing based on mathematical logic and Boolean algebra.

A VNA computer is centric by the bus and characterized by the all purpose memory for both data and instructions. A VNA machine is an extended Turing machine (TM), where the power and functionality of all components of TM including the control unit (with wired instructions), the tape (memory), and the head of I/O, are greatly enhanced and extended with more powerful instructions and I/O capacity.

Definition 24. A *Wang Architecture* (WA) of computers, known as the *Cognitive Machine* as shown in Figure 8, is a parallel structure encompassing an Inference Engine (IE) and a Perception Engine (PE) (Wang, 2006b, g), that is:

$WA \triangleq (IE \| PE)$
= (KMU// The knowledge manipulation unit

Figure 8. The architecture of a cognitive machine
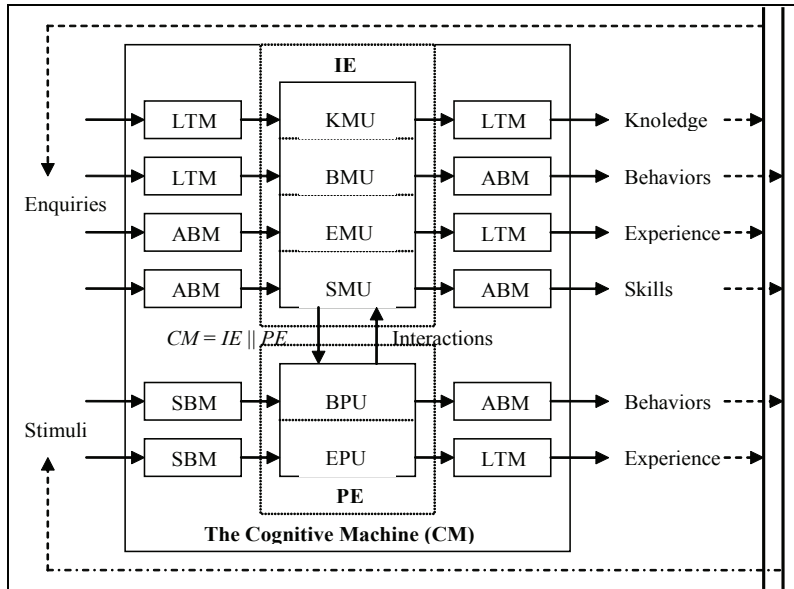
|| BMU//The behavior manipulation unit

|| EMU // The experience manipulation unit

|| SMU// The skill manipulation unit

)

|| (BPU   // The behavior perception unit

EPU // The experience perception unit

)

$$(21)$$

As shown in Figure 8 and Equation 21, WA computers are not centered by a CPU for data manipulation as the VNA computers do. The WA computers are centered by the concurrent IE and PE for cognitive learning and autonomic perception based on abstract concept inferences and empirical stimuli perception. The IE is designed for concept/knowledge manipulation according to concept algebra (Wang, 2006e), particularly the nine concept operations for knowledge acquisition, creation, and manipulation. The PE is designed for feeling and perception processing according to RTPA (Wang, 2002b) and the formally described cognitive process models of the perception layers as defined in the LRMB model (Wang et al., 2006).

Definition 25. *Cognitive computers* with WA are aimed at cognitive and perceptive *concept/ knowledge* processing based on contemporary *denotational mathematics*, that is, CA, RTPA, and SA.

As that of mathematical logic and Boolean algebra are the mathematical foundations of VNA computers. The mathematical foundations of WA computers are based on denotational mathematics (Wang, 2006b, c). As described in the LRMB reference model (Wang et al., 2006), since all the 37 fundamental cognitive processes of human brains can be formally described in CA and RTPA (Wang, 2002b, 2006e). In other words, they are simulatable and executable by the WA-based

Table 3. Classification of computing systems

		Behavior (O)	
		Constant	**Variable**
Event (I)	**Constant**	Routine	Adaptive
	Variable	Algorithmic	Autonomic
Type of behavior		*Deterministic*	*Nondeterministic*

cognitive computers.

Estimation of the Capacity of Human Memory

Despite the fact that the number of neurons in the brain has been identified in cognitive and neural sciences, the magnitude of human memory capacity is still unknown. According to the OAR model, a recent discovery in CI is that the upper bound of memory capacity of the human brain is in the order of $10^{8,432}$ bits (Wang et al., 2003). The determination of the magnitude of human memory capacity is not only theoretically significant in CI, but also practically useful to unveil the human potential, as well as the gaps between the natural and machine intelligence. This result indicates that the next generation computer memory systems may be built according to the OAR model rather than the traditional container metaphor, because the former is more powerful, flexible, and efficient to generate a tremendous memory capacity by using limited number of neurons in the brain or hardware cells in the next generation computers.

Autonomic Computing

The approaches to implement intelligent systems can be classified into those of biological organisms, silicon automata, and computing systems. Based on CI studies, *autonomic computing* (Wang, 2004) is proposed as a new and advanced computing technique built upon the routine, algorithmic, and adaptive systems as shown in Table 3.

The approaches to computing can be classified into two categories known as imperative and autonomic computing. Corresponding to these, computing systems may be implemented as imperative or autonomic computing systems.

Definition 26. An *imperative computing system* is a passive system that implements deterministic, context-free, and stored-program controlled behaviors.

Definition 27. An *autonomic computing system* is an intelligent system that autonomously carries out robotic and interactive actions based on goal- and event-driven mechanisms.

The imperative computing system is a traditional passive system that implements deterministic, context-free, and stored-program controlled behaviors, where a behavior is defined as a set of observable actions of a given computing system. The autonomic computing system is an active system that implements nondeterministic, context-dependent, and adaptive behaviors, which do not rely on instructive and procedural information, but are dependent on internal status and willingness that formed by long-term historical events and current rational or emotional goals.

The first three categories of computing techniques as shown in Table 3 are imperative. In contrast, the autonomic computing systems are an active system that implements nondeterministic, context-sensitive, and adaptive behaviors. Autonomic computing does not rely on imperative and procedural instructions, but

Table 4. Types of cognitive information

		Type of Output		Ways of Acquisition
		Abstract Concept	Empirical Action	
Type of Input	**Abstract Concept**	Knowledge	Behavior	*Direct or indirect*
	Empirical Action	Experience	Skill	*Direct only*

are dependent on perceptions and inferences based on internal goals as revealed in CI.

Cognitive Properties of Knowledge

Almost all modern disciplines of science and engineering deal with information and knowledge. According to CI theories, cognitive information may be classified into four categories known as *knowledge, behaviors, experience,* and *skills* as shown in Table 4.

Definition 28. The taxonomy of *cognitive information* is determined by its types of inputs and outputs to and from the brain during learning and information processing, where both inputs and outputs can be either abstract information (concept) or empirical information (actions).

It is noteworthy that the approaches to acquire knowledge/behaviors and experience/skills are fundamentally different. The former may be obtained either directly based on hands-on activities or indirectly by reading, while the latter can never be acquired indirectly.

According to Table 4, the following important conclusions on information manipulation and learning for both human and machine systems can be derived.

Theorem 9. The *principle of information acquisition* states that there are four sufficient categories of learning known as those of knowledge, behaviors, experience, and skills.

Theorem 9 indicates that learning theories and their implementation in autonomic and intelligent systems should study all four cat-

egories of cognitive information acquisitions, particularly behaviors, experience, and skills rather than only focusing on knowledge.

Corollary 3. All the four categories of information can be acquired directly by an individual.

Corollary 4. Knowledge and behaviors can be learnt indirectly by inputting abstract information, while experience and skills must be learned directly by hands-on or empirical actions.

The above theory of CI lays an important foundation for learning theories and pedagogy (Wang, 2004, 2006e). Based on the fundamental work, the IE and PE of cognitive computers working as a virtual brain can be implemented on WA-based cognitive computers and be simulated on VNA-based conventional computers.

Simulation of Human Cognitive Behaviors using the Contemporary Mathematics

The contemporary denotational mathematics as described in The Denotational Mathematics for CI section, particularly CA and RTPA, may be used to simulate the cognitive processes of the brain as modeled in LRMB (Wang et al., 2006). Most of the 37 cognitive processes identified in LRMB, such as the learning (Wang, 2006e), reasoning (Wang, 2006b), decision making (Wang et al., 2004), and comprehension (Wang & Gafurov, 2003) processes, have been rigorously modeled and described in RTPA and CA. Based on the

fundamental work, the inference engineering and perception engine of a virtual brain can be implemented on cognitive computers or be simulated on conventional computers. In the former case, a working prototype of a fully autonomic computer will be realized on the basis of CI theories.

Agent Systems

Definition 29. A *software agent* is an intelligent software system that autonomously carries out robotic and interactive applications based on goal-driven mechanisms (Wang, 2003c).

Because a software agent may be perceived as an application-specific virtual brain (see Theorem 3), behaviors of an agent are mirrored human behaviors. The fundamental characteristics of agent-based systems are autonomic computing, goal-driven action-generation, knowledge-based machine learning. In recent CI research, *perceptivity* is recognized as *the sixth sense* that serves the brain as the thinking engine and the kernel of the natural intelligence. Perceptivity implements self-consciousness inside the abstract memories of the brain. Almost all cognitive life functions rely on perceptivity such as consciousness, memory searching, motivation, willingness, goal setting, emotion, sense of spatiality, and sense of motion. The brain may be stimulated by external and internal information, which can be classified as willingness-driven (internal events such as goals, motivation, and emotions), event-driven (external events), and time-driven (mainly external events triggered by an external clock). Unlike a computer, the brain works in two approaches: the internal willingness-driven processes, and the external event- and time-driven processes. The external information and events are the major sources that drive the brain, particularly for conscious life functions.

Recent research in CI reveals that the foundations of agent technologies and autonomic computing are CI, particularly goal-driven action generation techniques (Wang, 2003c). The LRMB model (Wang et al., 2006) described in the Layered Reference Model of the Brain section may be used as a reference model for agent-based technologies. This is a fundamental view toward the formal description and modeling of architectures and behaviors of agent systems, which are created to do something repeatable in context, to extend human capability, reachability, and/or memory capacity. It is found that both human and software behaviors can be described by a 3-dimensional representative model comprising *action*, *time*, and *space*. For agent system behaviors, the three dimensions are known as *mathematical operations*, *event/process timing*, and *memory manipulation* (Wang, 2006g). The 3-D behavioral space of agents can be formally described by RTPA that serves as an expressive mathematical means for describing thoughts and notions of dynamic system behaviors as a series of actions and cognitive processes.

CI Foundations of Software Engineering

Software is an intellectual artifact and a kind of instructive information that provides a solution for a repeatable computer application, which enables existing tasks to be done easier, faster, and smarter, or which provides innovative applications for the industries and daily life. Large-scale software systems are highly complicated systems that have never been handled or experienced precedent by mankind.

The fundamental cognitive characteristics of software engineering have been identified as follows (Wang, 2006g):

- The inherent complexity and diversity
- The difficulty of establishing and stabilizing requirements
- The changeability or malleability of system behavior
- The abstraction and intangibility of software products
- The requirement of varying problem domain

knowledge
- The non-deterministic and polysolvability in design
- The polyglotics and polymorphism in implementation
- The dependability of interactions among software, hardware, and human beings

The above list forms a set of fundamental constraints for software engineering, identified as the cognitive constraints of intangibility, complexity, indeterminacy, diversity, polymorphism, inexpressiveness, inexplicit embodiment, and unquantifiable quality measures (Wang, 2006g).

A set of psychological requirements for software engineers has been identified, such as: (a) Abstract-level thinking; (b) Imagination of dynamic behaviors with static descriptions; (c) Organization capability; (d) Cooperative attitude in team work; (e) Long-period focus of attentions; (f) Preciseness; (g) Reliability; and (h) Expressive capability in communication.

Deductive Semantics of Software

Deduction is a reasoning process that discovers new knowledge or derives a specific conclusion based on generic premises such as abstract rules or principles. In order to provide an algebraic treatment of the semantics of program and human cognitive processes, a new type of formal semantics known as deductive semantics is developed (Wang, 2006f, g).

Definition 30. *Deductive semantics* is a formal semantics that deduces the semantics of a program from a generic abstract semantic function to the concrete semantics, which are embodied onto the changes of status of a finite set of variables constituting the semantic environment of computing (Wang, 2006g).

Theorem 10. The *semantics of a statement p, θ(p)*, on a given semantic environment Θ in deductive semantics is a double partial differential of the semantic function, $f_\theta(p) =$

$$f_p : T \times S \to V = v_p(t,s), t \in T \land s \in S \land v_p \in V \quad ,$$

on the sets of variables S and executing steps T, that is:

$$\theta(p) = \frac{\partial^2}{\partial t\, \partial s} f_q(p) = \frac{\partial^2}{\partial t\, \partial s} v_p(t,s)$$

$$= \mathop{R}_{i=0}^{\#T(p)} \mathop{R}_{j=1}^{\#S(p)} v_p(t_i, s_j)$$

$$= \mathop{R}_{i=0}^{1} \mathop{R}_{j=1}^{\#\{s_1, s_2, ..., s_m\}} v_p(t_i, s_j)$$

$$= \begin{pmatrix} & \mathbf{s_1} & \mathbf{s_2} & \cdots & \mathbf{s_m} \\ \mathbf{t_0} & v_{01} & v_{02} & \cdots & v_{0m} \\ (\mathbf{t_0}, \mathbf{t_1}] & v_{11} & v_{12} & \cdots & v_{1m} \end{pmatrix} \qquad (22)$$

where t denotes the discrete time immediately before and after the execution of p during (t_0, t_1], and # is the *cardinal calculus* that counts the number of elements in a given set, that is $n = \#T(p)$ and $m = \#S(p)$.

The first partial differential in Equation 22 selects all related variable $S(p)$ of the statement p from Θ. The second partial differential selects a set of discrete steps of p's execution $T(p)$ from Θ. According to Theorem 10, the semantics of a statement can be reduced onto a semantic function that results in a 2-D matrix with the changes of values for all variables over time along program execution.

Deductive semantics perceives that the carriers of software semantics are a finite set of variables declared in a given program. Therefore, software semantics can be reduced onto the changes of values of these variables. The deductive mathematical models of semantics and the semantic environment at various composing levels of systems are formally described. Properties of software semantics and relationships between the software behavioral space and the semantic environment are discussed. Deductive semantics is applied in the formal definitions and explanations of the semantic rules of a comprehensive

Table 5. Calibrated cognitive weights of BCSs

BCS	RTPA Notation	Description	Calibrated cognitive weight
1	$\rightarrow$	Sequence	1
2	|	Branch	3
3	|… |…	Switch	4
4	R^i	For-loop	7
5	R^*	Repeat-loop	7
6	R^*	While-loop	8
7	$\rightarrowtail$	Function call	7
8	$\circlearrowleft$	Recursion	11
9	|| or □	Parallel	15
10	↯	Interrupt	22

set of software static and dynamic behaviors as modeled in RTPA. Deductive semantics can be used to define abstract and concrete semantics of software and cognitive systems, and facilitate software comprehension and recognition by semantic analyses.

Cognitive Complexity of Software

The estimation and measurement of functional complexity of software are an age-long problem in software engineering. The cognitive complexity of software (Wang, 2006j) is a new measurement for cross-platform analysis of complexities, sizes, and comprehension effort of software specifications and implementations in the phases of design, implementation, and maintenance in software engineering. This work reveals that the cognitive complexity of software is a product of its architectural and operational complexities on the basis of deductive semantics and the abstract system theory. Ten fundamental basic control structures (BCSs) are elicited from software architectural/behavioral specifications and descriptions. The cognitive weights of those BCSs are derived and calibrated via a series of psychological experiments. Based on this work, the cognitive complexity of software systems can be rigor-

ously and accurately measured and analyzed. Comparative case studies demonstrate that the cognitive complexity is highly distinguishable in software functional complexity and size measurement in software engineering.

On the basis of the ERM model described in Theorem 5 and the deductive semantics of software presented in The deductive semantics of software section, the finding on the cognitive complexity of software is obtained as follows.

Theorem 11. The sum of the cognitive weights of all r_{ij}, $w(r_{ij})$, in the ERM model determines the operational complexity of a software system C_{op}, that is:

$$C_{op} = \sum_{i=1}^{n-1} w(r_{ij}), \, j = i+1 \qquad (23)$$

A set of psychological experiments has been carried out in undergraduate and graduate classes in software engineering. Based on 126 experiment results, the equivalent cognitive weights of the 10 fundamental BCSs are statistically calibrated as summarized in Table 5 (Wang, 2006j), where the relative cognitive

Table 6. Measurement of software system complexities

System	Time complexity $(C_t$ [OP])	Cyclomatic complexity $(C_m$ [-])	Symbolic complexity $(C_s$ [LOC])	Cognitive complexity		
				Operational complexity $(C_{op}$ [F])	Architectural complexity $(C_a$ [O])	Cognitive complexity $(C_c$ [FO])
IBS (a)	ε	1	7	13	5	65
IBS (b)	O(n)	2	8	34	5	170
MaxFinder	O(n)	2	5	115	7	805
SIS_Sort	O(m+n)	5	8	163	11	1,793

weight of the sequential structures is assumed one, that is, $w_1 = 1$.

According to deductive semantics, the complexity of a software system, or its semantic space, is determined not only by the number of operations, but also by the number of data objects.

Theorem 12. The *cognitive complexity* $C_c(S)$ of a software system S is a product of the operational complexity $C_{op}(S)$ and the architectural complexity $C_a(S)$, that is:

$$C_c(S) = C_{op}(S) \bullet C_a(S)$$
$$= \{\sum_{k=1}^{n_C} \sum_{i=1}^{\#(C_s(C_k))} w(k,i)\} \bullet$$
$$\{\sum_{k=1}^{n_{CLM}} \text{OBJ}(CLM_k) + \sum_{k=1}^{n_C} \text{OBJ}(C_k)\} \quad \text{[FO]}$$
$$(24)$$

Based on Theorem 12, the following corollary can be derived.

Corollary 5. The cognitive complexity of a software system is proportional to both its operational and structural complexities. That is, the more the architectural data objects and the higher the operational complicity onto these objects, the larger the cognitive complexity of the system.

Based on Theorem 11, the cognitive com-

plexities of four typical software components (Wang, 2006j) have been comparatively analyzes as summarized in Table 6. For enabling comparative analyses, data based on existing complexity measures, such as *time, cyclomatic,* and *symbolic* (LOC) complexities, are also contrasted in Table 6.

Observing Table 6 it can be seen that the first three traditional measurements cannot actually reflect the real complexity of software systems in software design, representation, cognition, comprehension, and maintenance. It is found that (a) Although four example systems are with similar symbolic complexities, their operational and functional complexities are greatly different. This indicates that the symbolic complexity cannot be used to represent the operational or functional complexity of software systems. (b) The symbolic complexity (LOC) does not represent the throughput or the input size of problems. (c) The time complexity does not work well for a system where there are no loops and dominate operations, because in theory that all statements in linear structures are treated as zero in this measure no matter how long they are. In addition, time complexity cannot distinguish the real complexities of systems with the same asymptotic function, such as in Case 2 (IBS (b)) and Case 3 (Maxfinder). (d) The cognitive complexity is an ideal measure of software functional complexities and sizes, because it represents the real semantic complexity

by integrating both the operational and architectural complexities in a coherent measure. For example, the difference between IBS(a) and IBS(b) can be successfully captured by the cognitive complexity. However, the symbolic and cyclomatic complexities cannot identify the functional differences very well.

CONCLUSIONS

This article has presented an intensive survey of the recent advances and ground breaking studies in *Cognitive informatics*, particularly its theoretical framework, denotational mathematics, and main application areas. CI has been described as a new discipline that studies the natural intelligence and internal information processing mechanisms of the brain, as well as processes involved in perception and cognition. CI is a new frontier across disciplines of computing, software engineering, cognitive sciences, neuropsychology, brain sciences, and philosophy in recent years. It has been recognized that many fundamental issues in knowledge and software engineering are based on the deeper understanding of the mechanisms of human information processing and cognitive processes.

A coherent set of theories for CI has been described in this article, such as the Information-Matter-Energy model, Layered Reference Model of the Brain, the OAR model of information representation, Natural Intelligence vs. Artificial Intelligence, Autonomic Computing vs. imperative computing, CI laws of software, mechanisms of human perception processes, the cognitive processes of formal inferences, and the formal knowledge system. Three contemporary mathematical means have been created in CI known as the *denotational mathematics*. Within the new forms of denotational mathematical means for CI, *Concept Algebra* has been designed to deal with the new abstract mathematical structure of concepts and their representation and manipulation in learning and knowledge engineering. *Real-Time Process Algebra* has been developed as an expressive, easy-to-comprehend, and language-independent notation system, and a specification and re-

finement method for software system behaviors description and specification. *System Algebra* has been created to the rigorous treatment of abstract systems and their algebraic relations and operations.

A wide range of applications of CI has been identified in multidisciplinary and transdisciplinary areas, such as the architecture of future generation computers, estimation the capacity of human memory, autonomic computing, cognitive properties of information, data, knowledge, and skills in knowledge engineering, simulation of human cognitive behaviors using descriptive mathematics, agent systems, CI foundations of software engineering, deductive semantics of software, and cognitive complexity of software systems.

ACKNOWLEDGMENT

REFERENCES

Bell, D. A. (1953). *Information theory*. London: Pitman.

Ganter, B., & Wille, R. (1999). *Formal concept analysis* (pp. 1-5). Springer.

Hoare, C. A. R. (1985). *Communicating sequential processes.* Prentice Hall.

Jordan, D. W., & Smith, P. (1997). *Mathematical techniques: An introduction for the engineering, physical, and mathematical sciences* (2nd ed.). Oxford, UK: Oxford University Press.

Klir, G. J. (1992). *Facets of systems science.* New York: Plenum.

Milner, R. (1989). *Communication and concurrency*. Englewood Cliffs, NJ: Prentice Hall.

Quillian, M. R. (1968). Semantic memory. In M. Minsky (Ed.), *Semantic information processing*. Cambridge, MA: MIT Press.

Shannon, C. E. (1948). A mathematical theory

of communication. *Bell System Technical Journal, 27*, 379-423, 623-656.

von Bertalanffy, L. (1952). *Problems of life: An evolution of modern biological and scientific thought*. London: C. A. Watts.

von Neumann, J. (1946). The principles of large-scale computing machines. Reprinted in *Annals of History of Computers, 3*(3), 263-273.

Wang, Y. (2002, August). On cognitive informatics (Keynote Speech). In *Proceedings of the 1st IEEE International Conference on Cognitive Informatics (ICCI'02)* (pp. 34-42), Calgary, Canada. IEEE CS Press.

Wang, Y. (2002). The real-time process algebra (RTPA). *The International Journal of Annals of Software Engineering, 14*, 235-274.

Wang, Y. (2003). Cognitive informatics: A new transdisciplinary research field. *Brain and Mind: A Transdisciplinary Journal of Neuroscience and Neurophilosophy, 4*(2), 115-127.

Wang, Y. (2003). On cognitive informatics. *Brain and Mind: A Transdisciplinary Journal of Neuroscience and Neurophilosophy, 4*(2), 151-167.

Wang, Y. (2003, August). Cognitive informatics models of software agent systems and autonomic computing (Keynote Speech). In *Proceedings of the International Conference on Agent-Based Technologies and Systems* (ATS'03) (p. 25), Calgary Canada. University of Calgary Press.

Wang, Y. (2003). Using process algebra to describe human and software system behaviors. *Brain and Mind: A Transdisciplinary Journal of Neuroscience and Neurophilosophy, 4*(2), 199–213.

Wang, Y. (2004, August). On autonomic computing and cognitive processes (Keynote Speech). In *Proceedings of the 3rd IEEE InternationalConference on Cognitive Informatics (ICCI'04)* (pp. 3-4), Victoria, Canada. IEEE CS Press.

Wang, Y. (2005, August). On the cognitive processes of human perceptions. In *Proceedings of the 4th IEEE International Conference on Cognitive Informatics (ICCI'05)* (pp. 203-211), Irvin, California. IEEE CS Press.

Wang, Y. (2005, May 1-4). On the mathematical laws of software. In *Proceedings of the 18th Canadian Conference on Electrical and Computer Engineering (CCECE'05)* (pp. 1086-1089), Saskatoon, Saskatchewan, Canada.

Wang, Y. (2005, August). The cognitive processes of abstraction and formal inferences. In *Proceedings of the 4th IEEE International Conference on Cognitive Informatics (ICCI'05)*, (pp. 18-26), Irvin, California. IEEE CS Press.

Wang, Y. (2006, May 8-10). A unified mathematical model of programs. In *Proceedings of the 19th Canadian Conference on Electrical and Computer Engineering (CCECE'06)* (pp. 2346-2349), Ottawa, Ontario, Canada.

Wang, Y. (2006, July). Cognitive informatics towards the future generation computers that think and feel. In *Proceedings of the 5th IEEE International Conference on Cognitive Informatics (ICCI'06)* (pp. 3-7), Beijing, China. IEEE CS Press.

Wang, Y. (2006, July). Cognitive informatics and contemporary mathematics for knowledge representation and manipulation (invited plenary talk). In *Proceedings of the 1st International Conference on Rough Set and Knowledge Technology (RSKT'06)* (pp. 69-78), Chongqing, China. Lecture Notes in Artificial Intelligence (LNAI) 4062. Springer.

Wang, Y. (2006, July). On abstract systems and system algebra. In *Proceedingsof the 5th IEEE International Conference on Cognitive Informatics (ICCI'06)*, (pp. 332-343), Beijing, China. IEEE CS Press.

Wang, Y. (2006, July). On concept algebra and knowledge representation. In *Proceedings of the 5th IEEE International Conference on Cognitive Informatics* (ICCI'06) (pp. 320-331), Beijing, China. IEEE CS Press.

Wang, Y. (2006). On the informatics laws and deductive semantics of software. *IEEE Transactions on Systems, Man, and Cybernetics (Part C), 36*(2), 161-171.

Wang, Y. (2006). *Software engineering foundations: A transdisciplinary and rigorous perspective* (CRC Book Series in Software Engineering 2). Boca Raton, FL: CRC Press.

Wang, Y. (2006, May). The OAR model for knowledge representation. In *Proceedings of the 19th IEEE Canadian Conference on Electrical and Computer Engineering* (CCECE'06) (pp. 1696-1699), Ottawa, Canada.

Wang, Y. (2006, July). On the Big-R notation for describing iterative and recursive behaviors. In *Proceedings of the 5th IEEE International Conference on Cognitive Informatics* (ICCI'06) (pp. 132-140), Beijing, China. IEEE CS Press.

Wang, Y. (2006, July). Cognitive complexity of software and its measurement. In *Proceedings of the 5th IEEE International Conference on Cognitive Informatics (ICCI'06)* (pp. 226-235), Beijing, China. IEEE CS Press.

Wang, Y., Dong, L., & Ruhe, G. (2004, July). Formal description of the cognitive process of decision making. In *Proceedings of the 3rd IEEE International Conference on Cognitive Informatics (ICCI'04)* (pp. 124-130), Victoria, Canada. IEEE CS Press.

Wang, Y., & Gafurov, D. (2003, August). The cognitive process of comprehension. In *Proceedings of the 2nd IEEE International Conference on Cognitive Informatics (ICCI'03)* (pp. 93-97). London: IEEE CS Press.

Wang, Y., Johnston, R., & Smith, M. (2002). Cognitive informatics. In *Proceedings of the 1st IEEE International Conference (ICCI02)*, Calgary, Alberta, Canada. IEEE CS Press.

Wang, Y., & Kinsner, W. (2006, March). Recent advances in cognitive informatics. *IEEE Transactions on Systems, Man, and Cybernetics (Part C), 36*(2), 121-123.

Wang, Y., Liu, D., & Wang, Y. (2003). Discovering the capacity of human memory. *Brain and Mind: A Transdisciplinary Journal of Neuroscience and Neurophilosophy, 4*(2), 189-198.

Wang, Y., & Wang, Y. (2006, March). On cognitive informatics models of the brain. *IEEE Transactions on Systems, Man, and Cybernetics, 36*(2), 203-207.

Wang, Y., Wang, Y., Patel, S., & Patel, D. (2006, March). A layered reference model of the brain (LRMB). *IEEE Transactions on Systems, Man, and Cybernetics (Part C), 36*(2), 124-133.

Yingxu Wang (yingxu@ucalgary.ca) is professor of cognitive informatics and software engineering, director of International Center for Cognitive Informatics (ICfCI), and director of Theoretical and Empirical Software Engineering Research Center (TESERC) at the University of Calgary. He received a PhD in Software Engineering from The Nottingham Trent University, UK, in 1997, and a BSc in Electrical Engineering from Shanghai Tiedao University in 1983. He was a visiting professor in the Computing Laboratory at Oxford University during 1995, and has been a full professor since 1994. He is Editor-in-Chief of International Journal of Cognitive Informatics and Natural Intelligence (IJCiNi)*, Editor-in-Chief of* World Scientific Book Series on Cognitive Informatics*, and editor of* CRC Book Series in Software Engineering*. He has published over 280 papers and 10 books in software engineering and cognitive informatics, and won dozens of research achievement, best paper, and teaching awards in the last 28 years, particularly the IBC 21st Century Award for Achievement "in recognition of outstanding contribution in the field of Cognitive Informatics and Software Science."*