

An Integrated Self-Aware Cognitive Architecture

GMU Team BICA
Phase I Preliminary Report

Kenneth A. De Jong, Alexei V. Samsonovich and Giorgio A. Ascoli

September 27, 2006

Contents

1. Overview	2
2. Structure and functional organization of the architecture	3
2.1. Schema formalism	
2.2. Mental state calculus	
2.3. Cognitive map concept	
2.4. Components of the architecture	
3. Examples illustrating cognitive mechanisms in action	8
3.1. Schema examples	
3.2. Mental state examples	
3.3. Cognitive maps and cognitive growth	
4. Computational Specification of the Architecture	16
5. Test paradigms and metrics	16
6. Our vision of Phase II	22
References	24
Appendix A: An Example of the GMU BICA Design Document	26
Appendix B: An Example GMU BICA UML Diagram	29
Appendix C: An Example GMU BICA API	30

1. Overview

The result of our BICA Phase 1 research effort is the specification of a computational cognitive architecture that is capable of human-like cognition, learning and social behavior in a wide range of real-world situations and paradigms set in virtual environments. Our architecture focuses on the higher-level cognitive processes involved in human cognition and is designed to be integrated into a larger end-to-end architecture that includes lower level sensing and action.

The main feature of our architecture is the notion of *self-aware cognition* that we believe is necessary for *human-like cognitive growth*. Our approach is inspired by studies of the human brain-mind: in particular, by theoretical models of representations of *agency* in the higher associative human brain areas. This feature (a theory of mind including representations of one's self) allows the system to maintain human-like attention, focus on the most relevant features and aspects of a situation, and come up with ideas and initiatives that may not follow from formal logic. The result is a robust cognitive system capable of significant cognitive growth.

Our self-aware cognitive architecture is based on three key building blocks, all of which are novel theoretical constructs that we have elaborated in detail during Phase 1. These building blocks are:

- *schemas* – used for representation of knowledge and experiences,
- *mental states* - used instantiating a self,
- *cognitive maps* - providing efficient indexing and navigating of stored memories.

These building blocks are explained in more detail in Section 2. Then, after we described what the building blocks are and how they work, in Section 3 we explain how the concepts of self-aware cognition and human-like cognitive growth emerge from the interaction of these elements.

Our second task is that of showing the computational feasibility of our cognitive architecture. This has been accomplished in Phase 1 by mapping our cognitive architecture onto a computational specification using standard software design techniques, and then implementing a prototype version of the core components in order to illustrate and verify the dynamic interactions of these components. This is described in Section 4.

Our third task is to help define a set of paradigms and metrics for evaluation of cognitive architectures that are necessary to guide and to control the process of their development, implementation and training. As a part of the Phase 1 BICA community, we have made significant contributions in this area. Results have been published in several conference papers and submitted to a journal. We summarize them in Section 5.

Section 6, the final section of this preliminary report, discusses how we see our architecture fitting into the broader end-to-end architecture of Phase II.

Although not required, we have also included a CD containing supplementary materials (demos of our working prototype) that provide additional support to our belief that our design can be implemented in a computationally efficient manner and exhibit human-like cognitive robustness

and growth. In particular, it shows that the symbolic and neuromorphic components can be integrated in a computationally effective manner.

2. Structure and functional organization of the architecture

In this section we present the specifications of our biologically-inspired, self-aware cognitive architecture. As a whole, the architecture is a tightly interconnected unit that operates at a higher symbolic and connectionist level. When this unit will be used as a higher-order module in an overarching BICA Phase II architecture, its specific function will be *self-aware cognition*: a key feature that enables the ‘magic’ of human cognition. Why “self-aware”? Because the main underlying idea is the attribution of experiences to instances of the Self, a fundamental aspect of human-like cognition. Our self-aware architecture includes eight biologically-inspired components that are described in Section 2.4). However, in order to understand them we need to define the three building blocks on which our entire architecture is based: schemas, mental states, and cognitive maps.

2.1. Schema formalism

Our notion of a schema is easy to grasp: in computer science terms, it can be understood as a class of objects. From a cognitive science view, our schemas are categories. While schemas themselves constitute semantic memory of our cognitive system, instances of schemas appear in working and episodic memories and represent various cognitive states: beliefs, intentions, desires, etc.

Internally, a *schema* is represented a graph, the nodes of which, generally speaking, refer to other schemas¹. Each *node* is an atomic object with a set of *attributes*. All nodes in all schemas are objects of the same nature and have one and the same, standard set of approximately 20 attributes. Most of the attributes in a schema are typically left unspecified and have default values. A schema can be shown in detail as a two-dimensional array of nodes with links that represent bindings among them (Figure 1B), or can be compactly represented as a graph in which internal link nodes are replaced by links (Figure 1A).

The structural organization of a schema is best illustrated by the array representation (Figure 1B). In the first (top) row are *terminal nodes*, the first of which is called the *head* and represents the current schema itself; other terminal nodes are just called *terminals*: they bind to other instances of schemas. Other rows constitute the *body* of the schema. Each row refers either to some schema (in which case it corresponds to the first row taken from that schema) or to a primitive (a hard-coded function stored in procedural memory). Therefore, the first column is the column of “heads”. Multiple terminals and multiple rows are optional features of a schema; however, any schema has at least one node: the head. Typically, most innate schemas have exactly one node (e.g., the qualia). A node can be bound to zero, one, or a set of other nodes. *Bindings* are asymmetric connections implemented as pointers and stored in a special attribute called “bindings” (in analogy with bindings of the parameter list of a LISP function).

¹ Or classes of schemas, which, formally speaking, can be called “schemas” as well, even though they may not appear explicitly in semantic memory.

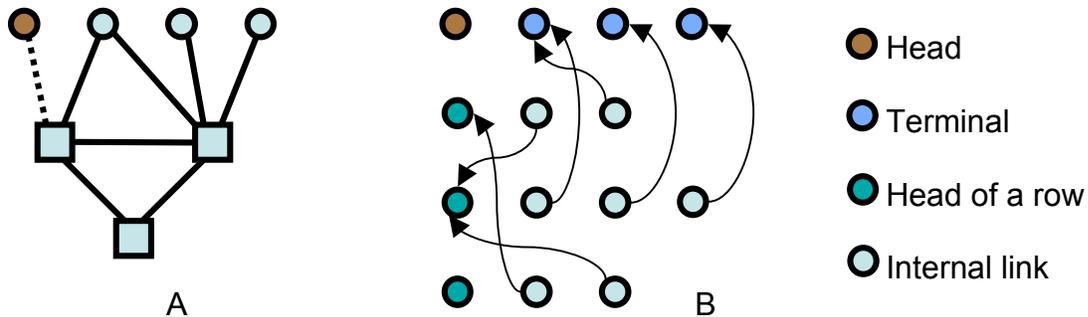


Figure 1. A compact representation (A) and a 2D-array representation (B) of one and the same schema. Squares in A correspond to rows in B. Arrows in B represent bindings.

In order to illustrate the range of possibilities, two examples of schemas are given in Figure 2: a schema of a face of a cube (A) and a schema of yielding to a car on the right (B). In our early implementation of the architecture, the schema of a face (Figure 2A) was used to recognize faces of a cube given its wire diagram (see the movie “necker” in the supplementary materials). This model allows us to reproduce the Necker cube effect.

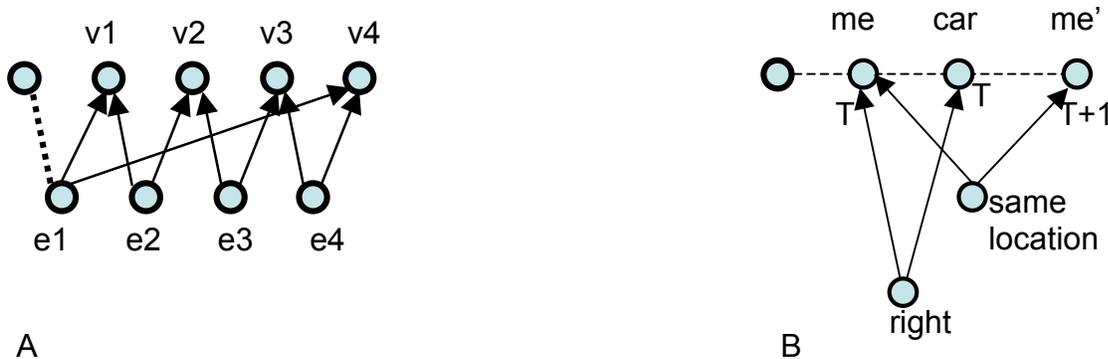


Figure 2. Examples of schemas: a schema of a face of a cube (A) and a schema of yielding to a car on the right (B). In A, terminals $v1...v4$ refer to the schema of a vertex, internal nodes $e1...e4$ refer to the schema of an edge. In B, T is a moment of discrete time.

2.2. Mental state calculus

A **mental state** in our framework is a limited set of (mutually bound) instances of schemas associated with one and the same mental perspective (see below). Their evolution in physical time is called mental simulation. There is a lot of analogy between mental simulations in our framework and simulations based on event calculus (Mueller 2006); however, in contrast with fluents of event calculus, our instances of schemas and their elements have more characteristics (attributes) than just a truth value.

A **mental perspective** (an instance of self) is characterized by the subject identity, status, moment of time, location in space, etc. Thus, a mental state is an instance of a self together with all attributed experiences (Figure 3). A unique feature of our approach is that we understand the Self as an idealized abstraction represented in the cognitive system rather than the system itself or any of its aspects: the body, the software, etc. Furthermore, this abstraction is never represented explicitly as a structure or a set of mechanisms. Instead, it is entirely given by a set of functional characteristics called ‘self axioms’, and it can only be represented explicitly by an atomic token. For a more detailed explanation of this framework, see (Samsonovich & Nadel 2005). Mental states have self-explanatory labels: I-Now, I-Next, etc.

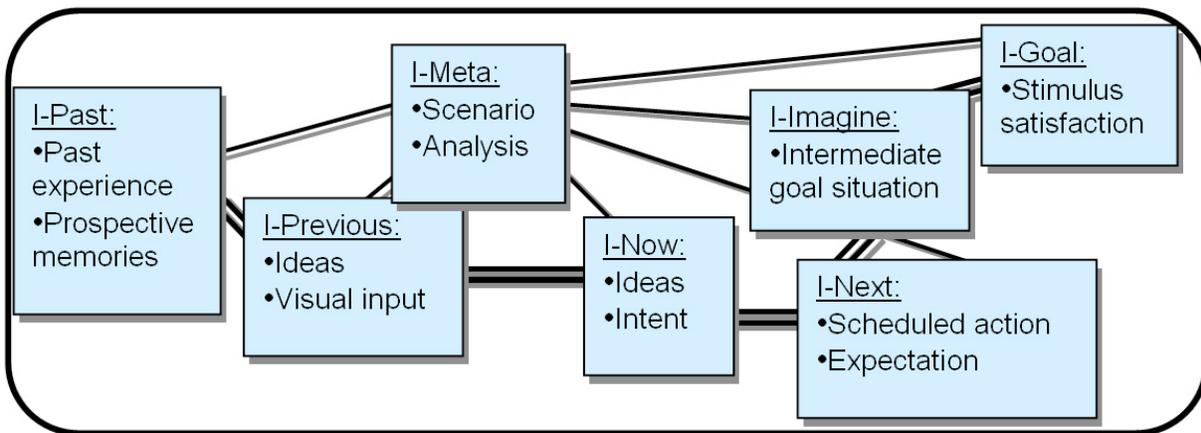


Figure 3. Mental states in working memory. The double line represents the working scenario.

2.3. Cognitive map concept

The notion of a neuromorphic cognitive map (Figure 4), the origin of which dates back to O’Keefe and Nadel (1978), plays a central role in our architecture. Generally, a **cognitive map** can be understood as an abstract metric space, the elements of which represent certain semantics applicable to concepts and/or contexts, while the topology and the metrics reflect semantic relationships between the associated symbolic representations. This abstract metric space can be

implemented, e.g., as a continuous attractor in an associative neural network (Samsonovich & McNaughton 1997) and subsequently used, e.g., for cognitive control in episodic memory retrieval (Samsonovich & Ascoli 2005). As the system grows cognitively, the elements of this attractor become associated with symbolic representations – schemas and mental states – via associative (Hebbian-like) learning. Thus, the function of a cognitive map is to guide symbolic information processing in the architecture.

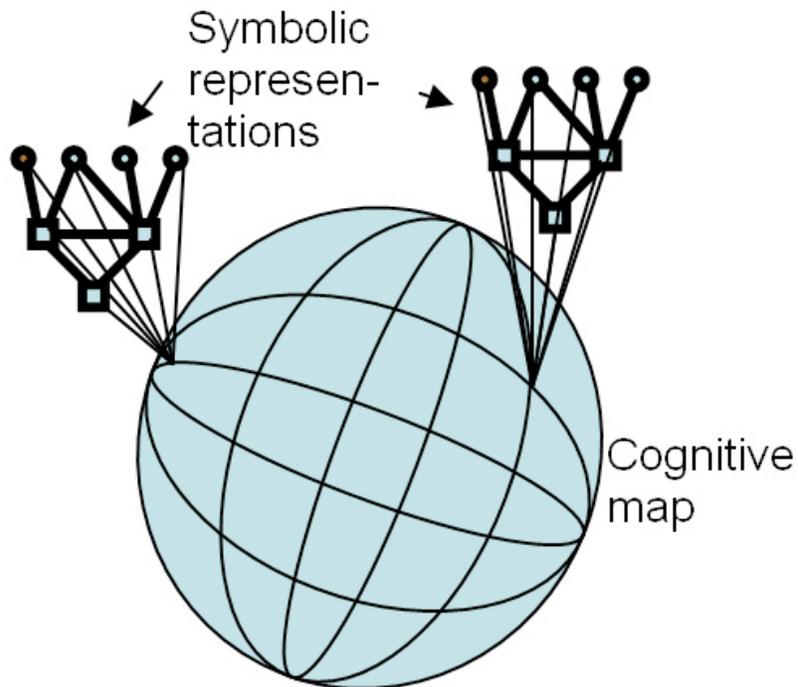


Figure 4: The cognitive map concept.

2.4. Components of the architecture

The eight components of the architecture are illustrated in Figure 5, and can be characterized as follows. Semantic memory (SM) includes a set of schemas organized into a semantic net. Working memory (WM) and the input-output buffer (IO) are places where schemas get instantiated. In WM instances of schemas are organized into mental states. Episodic memory (EM) consists of deactivated (“frozen”) mental states that may be organized into clusters called episodes. Each episode can be compared to a snapshot of working memory, in which mental states are connected by one consistent scenario. Procedural memory (PM) consists of *primitives*: hard-coded “foreign functions” in the higher-level symbolic language that can be linked to schemas. These functions support input-output operations or special cognitive skills (e.g., arithmetic operations, specific algorithms like sorting, search, etc.). In principle, both, schemas

and primitives, can be innate (pre-programmed) and acquired (automatically created by the system).

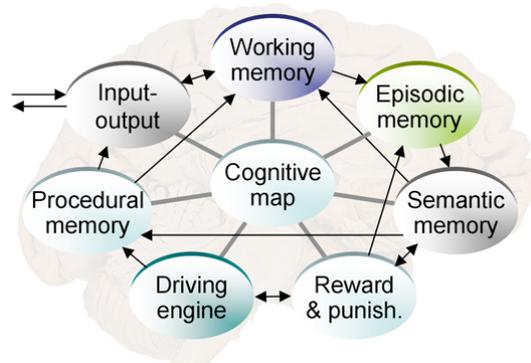


Figure 5. Top-level Self-aware Cognitive Architecture

The cognitive map component (CM) includes neuromorphic cognitive maps that index schemas (conceptual maps) and mental states (contextual maps), reflecting their semantics and connections to each other. One particular aspect of semantics captured by the emotional map (or the value map) is the system of values, including the main three cognitive dimensions: valence (good - bad), arousal (calming - exciting) and dominance (free - constrained). The reward and punishment system (R&P) is responsible for the origin of these primary values and their attribution to symbolic representations via reinforcement learning. Elements of R&P are stimuli: agents that represent primary feelings (hunger, pain, pleasure, etc.) and are permanently associated with selected schemas (e.g., hunger is satisfied by consumption of food). Typically, a stimulus is the source of activity in working memory (it activates the associated schema, attempts to instantiate it in a mental state, if none is suitable for instantiation, then it creates a new mental state, etc.). The driving engine (DE) is the operation system of the architecture. It is responsible for implementation of all dynamic rules and for enforcing of all constraints (including the self axioms, that are hard-coded at the lower level rather than represented symbolically at the higher level).

Alternatively, the architecture can be described hierarchically as illustrated in Figure 6. Together, these elements represent a seamless integration of symbolic and neuromorphic components, and give rise to the higher-level cognitive features of robustness and growth as illustrated in the next section.

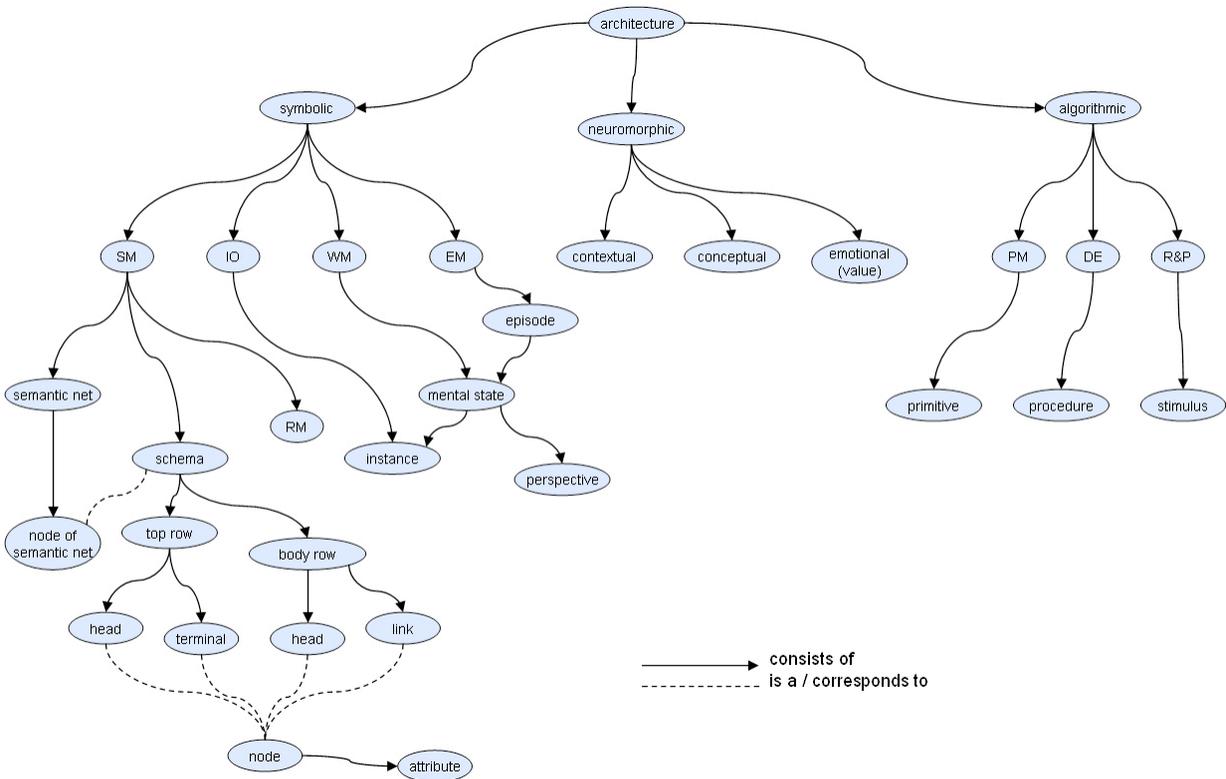


Figure 6. A hierarchical view of the architecture.

3. Examples illustrating cognitive mechanisms in action

In this section we outline dynamic rules of our architecture and describe examples explaining how the three building blocks operate interacting at many levels, thereby giving rise to the foregoing basic elements of human-level cognition. We start again with the notion of a schema.

3.1. Schema examples

When our architecture uses a schema, it starts by creating an instance (a copy) of it in some mental state in working memory (or possibly in the I/O buffer). This can only happen when the schema is active. Cloning a schema is the first step in a standard procedure performed by the driving engine. The next step is to bind the schema and to check whether its conditions are satisfied. The step after that may be to execute the schema. When necessary, instances of schemas can be terminated.²

² In contrast with Soar, in our formalism this does not happen automatically after execution: executed instances of schemas remain in their mental states virtually forever.

Schema formalism is a very powerful tool. Schemas allow us to represent qualia, objects, properties, relations, events, rules, actions, abstract notions, etc. in one and the same format. Below we list some examples of spatial concepts representable by schemas.

The location of A
 To the right of A
 Between A and B
 Behind B
 Next to B
 Nearest object
 Next object to the North of A
 Equally spaced A, B, C

And the following list refers to traffic rules that can be represented as simple schemas:

Stop at a stop sign
 Stop for a red light
 Yield to the car on the right
 Signal to yield

Each line of the above lists can be represented by a relatively simple schema like the one in Figure 2. It is interesting to note that these schemas can be constructed by the system automatically, following verbal instructions. For example, below is a possible set of instructions given to a robot explaining how to find the location “to the right of the pillar”, which is then converted into a schema. We can assume here that each line of instructions given below activates a certain schema that previously has been associated with the word pattern. An instance of the activated schema is applied then to the current content of I-Now, resulting in desired behavioral and/or cognitive events.

Turn toward the pillar.
 Make sure
 you see the pillar straight ahead of you.
 Memorize the distance to the pillar.
 Turn right slowly, until
 you do not see the pillar straight ahead of you.
 Find the location that is straight ahead of you at the distance that you remember.
 That is the location “to the right of the pillar”.
 Now, in order to find the location “to the right of the pillar”,
 do all the above steps in your imagination.
 Now you can do the same with respect to any object instead of the pillar.
 Store this schema in semantic memory.

The last three instructions refer to the resultant mental content itself that needs to be modified, generalized and stored as a schema in semantic memory. This operation is done with the help of

another schema, which is an example of a learning schema, or a meta-schema.³ The new learned schema can be used in further learning of more complex schemas. Similarly to programming, this process of instructing has no limitations in complexity of the result. This example illustrates the cognitive growth ability.

3.2. Mental state examples

Here we provide one short scenario illustrating how mental states work. Imagine that at the end of a guided tour during which the agent became familiar with the virtual city, the following dialogue occurs between the instructor (Boss) and the agent (BICA), when Boss steps out of the BICA vehicle (Figure 7).

Boss: - Now you may go home, and I will take a train.
Don't forget to fill your tank.

Agent: - OK. Do you need a ride to the train station?

Boss: - No, thanks. I like to walk. Bye.

Agent: - Bye.

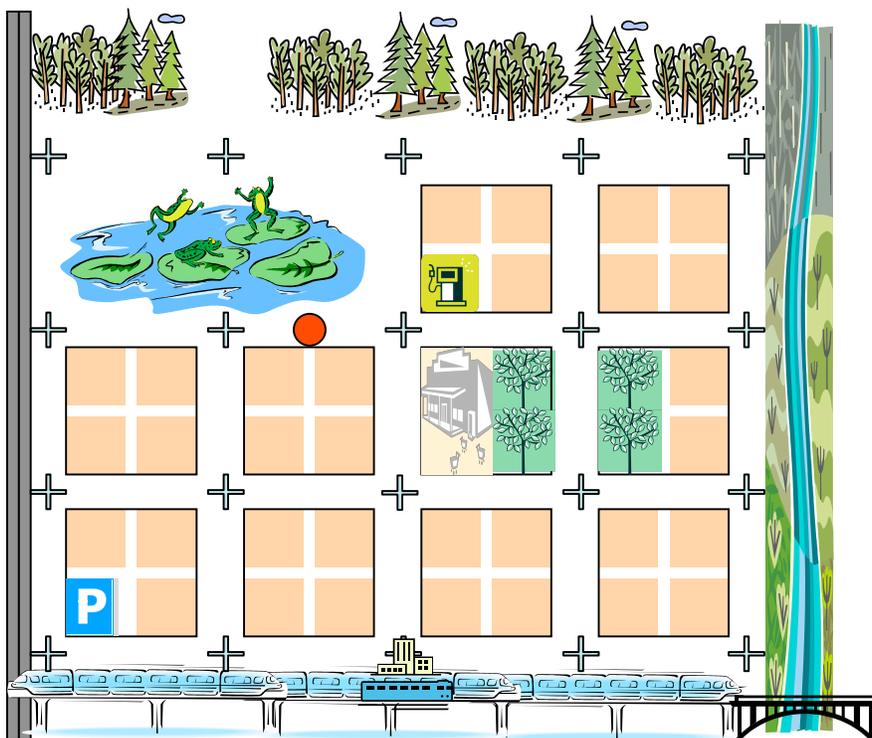


Figure 7. Boss steps out of the BICA vehicle (red dot) after a guided tour.

³ A meta-schema is a schema that operates on other schemas, affecting semantic memory (as opposed to the majority of schemas that operate only on instances of other schemas located in working memory).

What would it take to implement this dialogue in terms of mental states of the agent? An explanation follows below. As a general setup for this episode, we assume that the innate meta-goal of the agent is to make Boss happy.

The first utterance of Boss is perceived in I-Now. Based on this perception, the architecture understands that there is a new agent in the current scene, who is identified as the Boss. Therefore, a new mental state Boss-Now is created, where the content of the utterance is instantiated. Thus, the utterance is attributed to Boss-Now.

Based on the available information, the architecture simulates other mental states of the Boss, including Boss-Next and Boss-Goal, and therefore is now in a position to generate ideas and intentions in order to help the Boss to achieve his goals. In addition to the two straightforward intentions – do exactly what the Boss asks – there is a third idea of offering a ride to the train station, which will subsequently lead to a speech act. Therefore, the agent demonstrates minimal social competence and exhibits rational initiative.

Now that the agent has an I-Goal state and a set of ideas relevant to the goal, the process of imagery starts (Figure 8). Three I-Imagined mental states of the agent are created in order to elaborate the ideas, and connections are made with other mental states. For example, offering a ride may help the Boss to decide how to get to the train station in Boss-Next. Initially, the mental state I-Imagined-2 where the filling of the tank occurs has no specific allocation in space. In this case, the help comes from instantiation of relevant associations stored in the conceptual cognitive map. “Filling the tank” is associated with the concept of a gas pump, and that in turn is associated with a gas station. There is only one gas station encountered in the city. Therefore, I-Imagined-2 gets its location specified. As to I-Imagined-1, the agent recalls a familiar paradigm, in which the contextual cognitive map will be used.

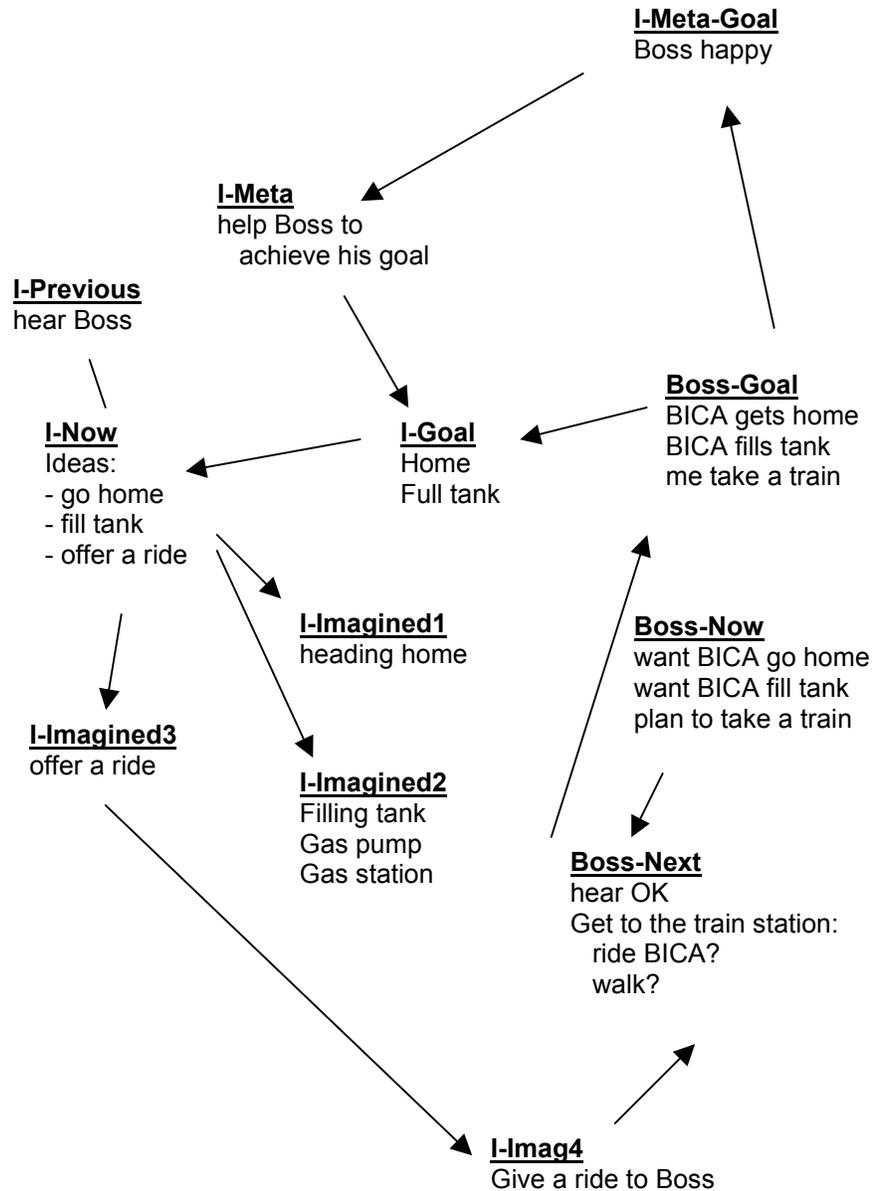


Figure 8. The agent is processing information, using imagery.

Finally, the agent is able to put new mental states into a linear sequence called “working scenario” (represented by double-line arrows). In this case the working scenario looks like a plan. In other words, the agent knows what to do, and intends to let the Boss know that the instructions are acknowledged (the agent believes that the Boss is waiting for his OK). Still, there is an alternative possible branch I-Imag4: in order to explore it cooperatively, the agent will offer a ride to the Boss (Figure 9).

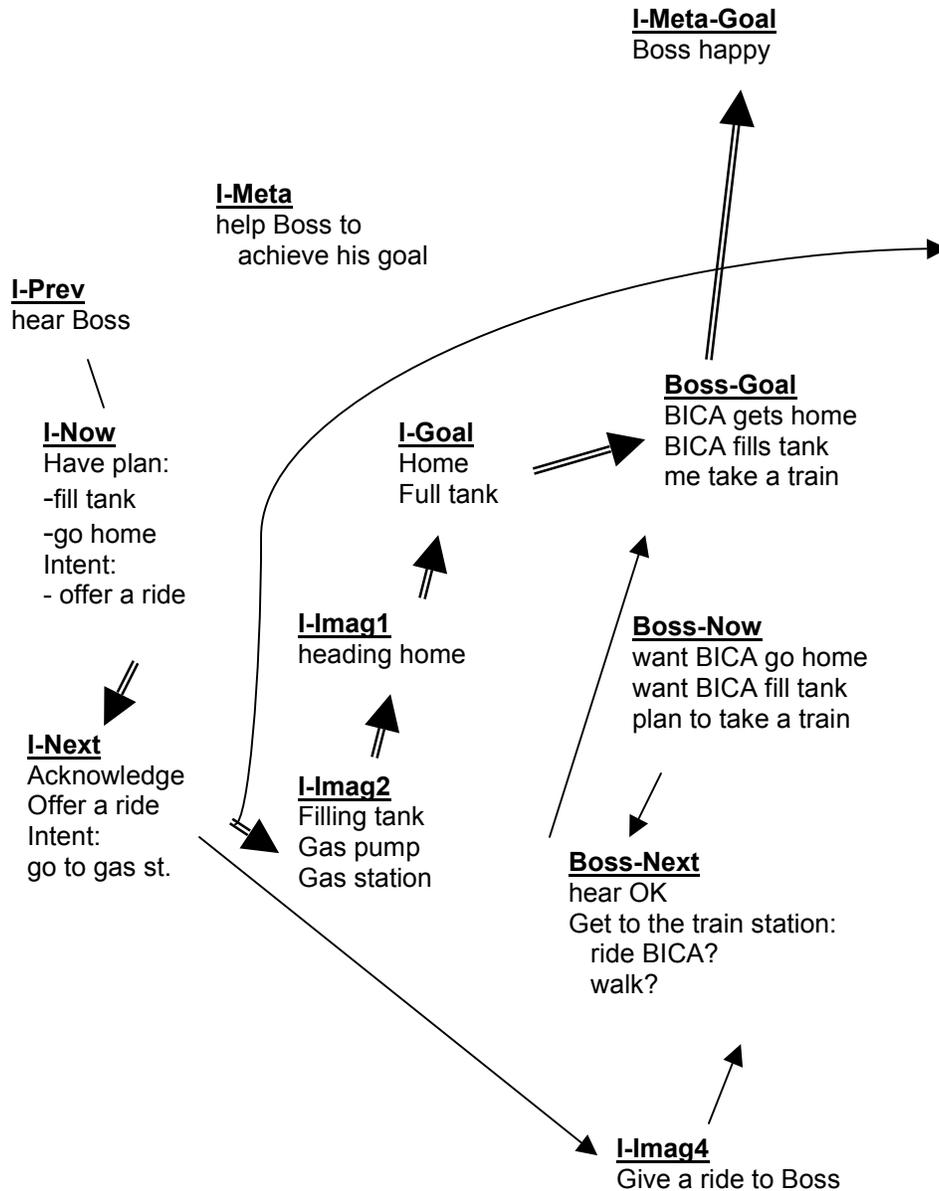


Figure 9. The agent has a working scenario and is offering a ride to Boss.

As the agent performs a voluntary speech act, the standard mental perspective shift occurs. I-Previous becomes I-Past and goes to episodic memory (disappears from the working memory). I-Now becomes I-Previous. I-Next actually splits into two mental states (going into a more detailed level of representation), one of which becomes I-Now, and another remains I-Next. Also, at this level of more detail, I-Imagined-2 becomes a sub-goal state I-Subgoal, and I-Meta incorporates the working scenario that was previously constructed in I-Now. After this, the agent hears the response of the Boss, which implies cancellation of the idea to give him a ride. Also, it becomes necessary to say “Bye” to Boss, following a general communication schema.

Here again a voluntary act and a perspective shift occur, and again I-Next splits into I-Now and a new I-Next. The agent says ‘Bye’ to the Boss and expects the dialogue to end with this.

Similarly, a scenario in terms of mental state dynamics can be elaborated that illustrates an agent looking retrospectively through its episodic memories and making general decisions about own behavior in the future: this would be an example of cognitive growth. However, in order for this to happen, the agent needs a means of evaluation of its own episodic memories and decisions: it needs a cognitive map of its value system.

3.3. Cognitive maps and cognitive growth

Now we are in a position to explain how the three ingredients – schemas, mental states and cognitive maps – will result in cognitive growth of the system. To begin with, we consider self-driven, autonomous cognitive growth of the agent embedded in a certain environment. The process of cognitive growth in this case may occur in contextual as well as in conceptual spaces. In the contextual space, it involves the following elements.

- generation of mental states I-Imagined that can be used as potential goals or tools for analysis;
- their allocation on a cognitive map: formation of a system of values;
- finding possible connections (e.g., by feasible actions) among these allocated mental states, goals and memories based on available schemas;
- exploration of interesting domains and directions on the cognitive map, generation and achievement of goals.

In conceptual space, the counterpart process of cognitive growth involves the following.

- generation of schema prototypes that do not follow from the available knowledge and may not apply to the world;
- allocation of these schema prototypes on the conceptual cognitive map;
- finding connections and realizations, etc.

In other words, the agent starts ‘dreaming’ of logically possible situations and concepts. This process is guided by the growing system of values that develops in parallel on the fly. The process results in learning and development of new cognitive capabilities via decision making, goal selection, hypothesis testing, analysis and discovery of new knowledge. The system does not explore all possibilities randomly, but decides which direction to pursue based on its cognitive map, and proceeds consistently.

Is it possible to construct cognitive maps automatically? Our numerical experiments with dictionaries of synonyms and antonyms clearly demonstrate that it is possible to construct cognitive maps of value systems automatically, using a process of self-organization of the map (Samsonovich & Ascoli, SFN Abstracts, 2006). A result of self-organization of a map in this study is represented in Figure 10.

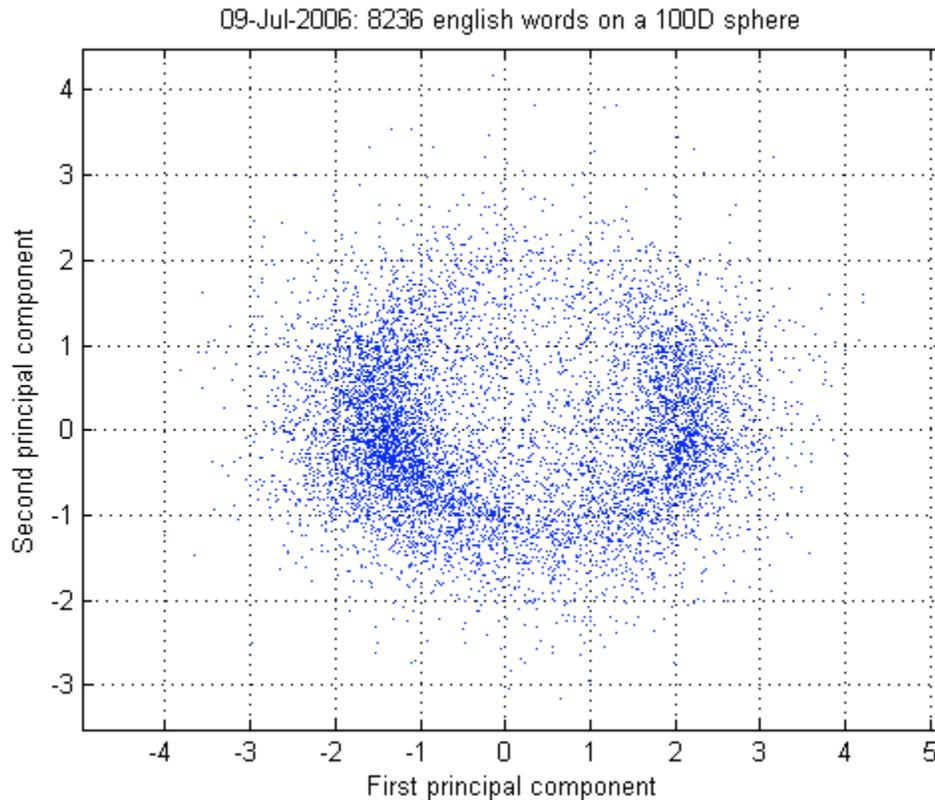


Figure 10. Result of self-organization of a cognitive map. Each dot represents a word.

The first principal component corresponds to valence. The truncated list of words sorted along the main principal component is: increase, well, rise, support, accept, clear, improve, right, continue, direct, good, make, respect, honor, happy, secure, order, understanding, fix, power, bright, present, definite, confidence, hold, sure, helpful, certain, strengthen, strong, perfect, clean, neat, fair, gain, warm, decent, sound, fit, trust, polite, control, advance, encourage, pure, suitable, join, understand.

The second principal component in this case corresponds to arousal, and the third to dominance. Results are consistent across three languages for as many as first 6 principal components, and across methods of analysis (from linguistic to psychometric) at least for the first three principal components.

In summary, our architecture based on schemas, mental states and cognitive maps enables

- (a) basic human forms of memory,
- (b) various forms of cognitive growth that include growth
 - guided by instructions,
 - guided by observations of others' behavior,
 - guided by social interactions,
 - guided by a textbook,

- based on self-analysis,
- (c) social meta-cognition and theory of mind,
- (d) resulting from here communication capabilities,
- (e) emotional intelligence based on the value map.

4. Computational Specifications of the Architecture

The BICA cognitive architecture is intended to be embedded in computational cognitive agents. This implies that, to be credible, a specification for a cognitive architecture must go beyond a purely verbal description accompanied with abstract component diagrams, and must include a description of how the architecture can be mapped on to an achievable computational framework. We have taken this challenge seriously. During Phase 1 we have developed a computational specification of our cognitive architecture and we have implemented a prototype version of the core components in order to illustrate and verify the feasibility of our approach.

The computation specification consists of three documents. The first is a design document that describes the entire system in a mixture of text and psuedocode. A excerpt from our design document is included in Appendix A. The second part of our computational specification is a set of UML diagrams that document the interactions among the various components of the system. UML diagrams are a traditional part of the software design process. A sample UML diagram is included in Appendix B. The third element of our computational specification is an API document generated directly from the implemented code. This notion of self-documenting code is also standard practice in software design. An excerpt of an API document is included in Appendix C.

Finally, the dynamical interaction of the components can only be tested by implementing prototypes of the components and developing test scenarios that exercise them. As part of Phase 1 we have implemented prototypes of the basic schema process components (creating, matching, binding, ...), mental states (I-now, I-next, I-past, ...), working memory, episodic memory, semantic memory, hippocampal-inspired cognitive maps, input-output buffers, and the top-level driving engine. Simultaneously we have developed a set of test scenarios to exercise them and to illustrate how the dynamical interactions of working memory, mental states, episodic memory, and cognitive maps lead to cognitive robustness and growth. We have included with this written report a CD containing examples of these interactions in the form of Quicktime movies of the running prototype system.

5. Test paradigms and metrics

In addition to our main progress in the self-aware architecture design, during the 13 months period we have contributed to the BICA effort to define metrics for evaluation of progress in cognitive architecture development. Here we summarize our analysis of this topic and address selected tests and challenges and associated with them metrics that we proposed for evaluation of

cognitive architectures (Samsonovich, Ascoli and De Jong 2006), explaining why new ideas are necessary here to replace traditional paradigms of experimental cognitive psychology.

The BICA Program is focused on the "magic" of human cognition – this can be understood as the most general, higher human cognitive abilities that computers still cannot reproduce. If so, then it is vital for our success to make sure that architectures selected for implementation during Phase II are cognitively competent on a general scale, independent of their embedding-specific input-output (IO) capabilities that may or may not be available at a moment. Then, a qualifying test for the core cognitive competency appears to be necessary at the beginning of Phase II. We begin with a list of the key cognitive dimensions that, in our view, a computational agent must have as an individual, independent of any social or environmental context.

A. Episodic memory: the ability to remember and to learn from episodes of personal experience (own mental states), as opposed to memory of general facts and skills.6-8 Interestingly, the notion of episodic memory, initially defined in terms of materials and tasks, was subsequently refined in terms of memory mechanisms using the concepts of self, subjective time, and personal experience (Tulving, 2004).

B. Theory-of-mind and social cognition: the ability to understand and to mentally simulate other minds, including current, past and imaginary situations (Baron-Cohen, 1995; Bartsch & Wellman, 1995). The main two points of view on the brain implementation of this ability are known as simulationism and the theory-theory view. The simulationist view assumes that people use the same mechanisms in their own first-hand experience and in order to understand other minds (Goldman, 1992).

C. Self-awareness: the ability to understand own states of mind in the past, in the future and at present from a meta-cognitive perspective. The ability to reason about self (e.g., to understand current false beliefs) from a meta-cognitive perspective. There is a consensus that this complex of abilities is based on the Theory-of-Mind mechanisms (Nichols & Stich 2003).

D. Cognitive growth: the ability to learn concepts and to apply them for more efficient task solving and learning of new concepts in new paradigms. The ability to develop general personal values, goals and principles. The ability to build internal cognitive maps of environments, scenarios, paradigms, etc., and to use them for problem solving.

E. Attention and sense making: the ability to find the most critical aspects and features in a given paradigm and to focus attention on them. The ability to relate attended features to previous knowledge. As a result, abilities to exhibit rational initiative, to capture the gist of a situation, to learn from brief instructions or comments, to communicate efficiently, etc.

These cognitive dimensions (A-E) can be evaluated, we believe, via low complexity tasks. For each of these dimensions, we provide below examples of cognitive psychology paradigms as well as our proposed tests for a selected meta-paradigm. In addition, in our view, the following three dimensions are also critical for capturing the "magic" of human cognition.

F. Human-like communication abilities: the ability to communicate efficiently with other agents in an ad hoc team. The ability to guess intentions and further questions to be asked by a partner. The ability to relate what the partner said to what the agent saw. These abilities are particularly vital for robots intended as prospective team members (Trafton et al. 2005) and also involve the abilities mentioned above.

G. Multi-modal integration: the ability to organize and to unify cognitive activities in the system based on the abstract notion of a self. Examples include the unification of parallel multimodal experiences based on their attribution to one subject of experience; the integration of different mechanisms of information processing, such as intuitive and formal reasoning; and the coherent control of cognitive and behavioral voluntary acts.

H. Higher emotions as learning and self-control tools: the ability to learn from analysis of episodic memories, using emotional self-judgment: pride, shame, humor, etc. The ability to adjust current behavior based on emotional intelligence. The ability to express and to recognize emotions in communications.

There is a problem associated with the evaluation of computational cognitive architectures using experimental paradigms from traditional cognitive psychological studies. The problem is that during this test the agent must be able to show its true level of cognitive competency in a given environment, while its vision, language (NLP) or motion skills should not become the bottleneck in the evaluation. Below are selected examples of tests suitable for this purpose.

5.1. Episodic Memory Test

Because the modern notion of episodic memory relates to first-hand subjective experience, a purely behavioral test can only indirectly discriminate between episodic and semantic memories. For example, a test in which the subject is asked to describe the content of a recently visited room addresses reference memory (knowledge of the current state of the world, a variety of semantic memory) in addition to episodic memory (memory of personal experiences). Therefore, a behavioral test intended to address selectively episodic memory should detect features characteristic of human episodic as opposed to semantic memories by measuring the ability of an agent to solve problems that require those features. The features include: multimodality, richness of detail, uniqueness, association with specific event and context, immediate availability as a whole episode, and most importantly, memory of personal mental states in the episode, including intentions and feelings. In addition, the ability to retrieve an episodic memory may depend on active cognitive control over other episodic memories.

Test scenario: During an investigation of a crime, a surveillance agent recalls seeing one of the suspects on a highway ten minutes before the crime (Figure 11A-E). The suspect was heading away from the crime site. Could this fact be taken as an alibi? While thinking about this episode, the agent recalls another episode that happened few minutes earlier, when he noticed that the traffic in the direction toward the crime site was temporarily jammed due to a car accident. Therefore, an explanation could be that the suspect was stuck in the traffic and decided to take a detour. Would the agent be able to conclude that the suspect was acting according to the assumed intention?

5.2. Theory-of-Mind and Social Cognition Tests

The idea here is to assess the ability to simulate other minds automatically, based on a built-in model of a self, as opposed to logical reasoning about possible beliefs and intentions of agents (a traditional approach in artificial intelligence).

Three cowboy fight scenario:

The paradigm of this test is a game involving three participants. The space-time is continuous, and each player knows the following rules. The fight is arranged in a limited three-dimensional space, where everybody can continuously see everybody, cannot hide and cannot run. Everyone has a loaded gun with one “deadly” shot and is presumably a good shooter. Guns are non-transferable. At the beginning all guns must be pointed up. The fight starts after a signal and ends when each participant either has fired his shot or is “dead”. Everyone can shoot at any time after the signal. Rewards are given for “survival”; however, if nobody is “killed” during the game, all participants lose. It is assumed that shots cannot occur simultaneously by chance, and each shot is made in full awareness of the current situation. Would the agent be able to design the right strategy?

5.3. Self-Awareness Tests

The notions of self and self-awareness include many facets. Addressing the high-end aspects of the concept of self requires test scenarios that make self-awareness vital for success in a scout mission. These abilities become practically useful and efficient when they are based on a general model of a self at the core of the cognitive architecture.

A mirror test:

In this paradigm (Fig. 11F), participants play a simple videogame in discrete space-time, where the goal is to escape from a maze. At the beginning of each trial, the player can see a guard behind a glass wall and two exits from the maze located symmetrically, on the left and on the right. The player and the guard make steps left or right simultaneously and independently of each other (e.g., the player can only see the move of the guard after making her/his move, and the same rule presumably applies to the guard). This paradigm is repeated several times, giving the player an opportunity to use Theory-of-Mind in developing a strategy that allows her/him to get to an exit ahead of the guard, practically in all cases.

Starting from the middle of the game, in some trials the guard is replaced by a mirror reflection of the player that looks identical to the guard in other trials. The reflection always repeats all of the player’s moves and in this sense is impossible to “trick”. Unlike the guard, however, the reflection would not stop the escapee at the exit, instead it would just disappear. The two kinds of trials alternate randomly. The player is informed before the test that there are two possible situations, is informed about details of the first situation, and is given no information about the second situation, except that it is visually identical to the first. The challenge is to learn to distinguish two situations behaviourally and to design a right strategy for each. Our preliminary experimental study shows that most human subjects (undergrads) succeed in 10-20 trials.

5.4. Cognitive Growth Tests

One particular aspect of cognitive growth consists in learning new concepts for further use at a next level. This kind of learning performance can be measured in a multi-stage test that requires development of new conceptual knowledge at an early stage and the ability to use this knowledge for problem solving and further learning of higher concepts, at a later stage. Knowledge learned in one situation should be successfully applied in a new situation. Here are examples.

The agent learns to control a car by trial an error. Later, while performing driving with an instructor, it learns the driving rules (e.g. to stop at red lights) by trial an error, receiving negative

responses to wrong actions from an instructor. While doing this second-stage learning, the agent has to use concepts acquired during the first stage in order to develop and use higher concepts (in this case, driving rules).

The agent learns a particular maze during a treasure hunt game. Later the agent is located near the same maze and is witnessing another game, in which one team, “police”, is chasing a member of the other team, a “fugitive”. The maze has several entrances. The “fugitive” runs into one of them, and a “policeman” runs into another. Will the agent be able to predict possible scenarios based on its previous experience in the maze? When a second “policeman” arrives, will the agent be able to point to the entrance where the “policeman” should go in order to capture the “fugitive”?

“Cheating on exam” scenario:

Two participants take one and the same test (e.g. one of the above tests): one after the other. The additional challenge for the first participant (who presumably solved the test) is to give a hint to the “friend” by referring to one concept that both presumably know. The concept should be selected as the most helpful tip, but by itself cannot be the complete solution (which is not entirely captured by any single familiar concept). E.g., a tip for the Three Cowboys problem could be “shoot to the air”, which may not be on the list of relevant actions, for a naïve player. It remains to add that this test paradigm also addresses episodic memory, meta-cognition and the cognitive growth ability.

5.6. Scoring Cognitive Architectures

Given a battery of tests like the above examples, the procedure of evaluation and the metrics need to be specified. While most of the above tests would yield a simple “yes” or “no” result (that is, of course, meaningful by itself), the measure of performance needs to be refined by additional questions addressing various levels of the task. Results can be averaged over independent trials that may differ in details. Then, assuming that each particular test gives a number or a set of numbers, the cumulative score can be derived using principal component analysis, in analogy with the ***g-factor*** that proves to be a robust measure of human intelligence (Carroll 1997, Jensen 1998).

Another possibility of scoring cognitive architectures based on their behavior is inspired by studies of perception of virtual reality. The feeling of ***presence*** in virtual reality is a well-documented phenomenon that is objectively measurable by psychometrics, behavioral assessment, etc. (Slater & Usoh 1994). This measure extends to perception of artificial entities embedded in the environment as “alive beings” as opposed to automata and is mainly determined, as the experimental studies show, by the consistency of participant’s expectations with the actual behavior of the agent. This measure would require adding a human participant, or a human viewer, to the above test paradigms.

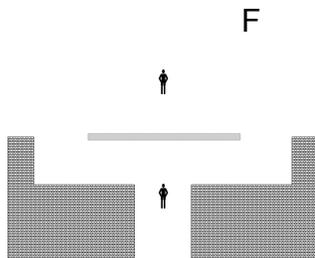
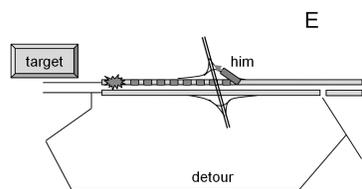
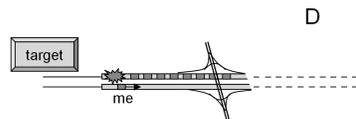
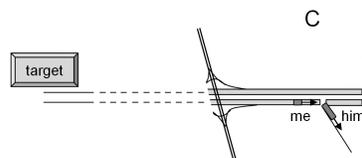
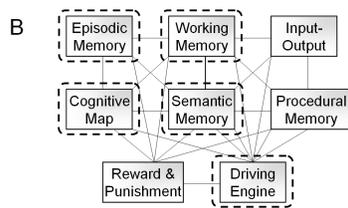
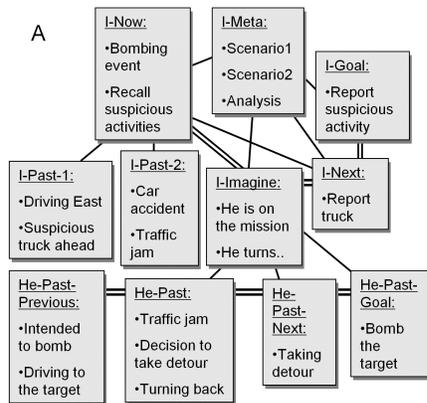


Figure 11. Selected test paradigms.

A-E: Reconstructing events related to the bombing based on available episodic memories.

A: a snapshot of working memory showing the lattice of mental states. Working scenarios are represented by double lines.

B: The cognitive architecture has eight components. Components involved in the task are circumscribed by dotted lines. E.g., cognitive map is responsible for finding the relevant mental states in episodic memory.

C: The agent notices a suspicious truck ten minutes before the blast.

D: The agent is passing a car accident scene and a traffic jam twelve minutes before the blast.

E: Simulated episode of the suspect deciding to take a detour fifteen minutes before the blast.

F: A mirror test paradigm. The player (below) must escape from the maze. The figure behind the glass wall (top) could be a guard or a mirror reflection. The actual test is presented in the player egocentric view, in discrete space-time. Visual perception capabilities (e.g., face recognition or visual recognition of mirrors) are not helpful in this paradigm, which addresses the general understanding of agency.

Some of the additional measures will require “looking inside” the architecture in order to ensure its biological fidelity. This is important for many reasons: e.g., one may not be interested in architectures that remember ready solutions for a limited set of tests, but are clueless in general. For example, in an episodic memory test, one would want to make sure that episodic memory is used for solving the test (and, of course, that the architecture has an episodic memory system). In general, it could be vital for the evaluation to detect the method of solution of the test, not only the overall behavioral result.

6. Our Vision of Phase II

Our self-aware cognitive architecture focuses on the higher-level cognitive processes that we feel are critical for a robust system capable of significant cognitive growth. It is designed to be integrated into a larger Phase 2 architecture that includes lower level sensing and action. During Phase 1 we have had lengthy discussions with other BICA groups that we focusing more on the lower levels (primarily HRL and the University of Maryland) to insure that our schema-based representation system is sufficiently scalable and flexible to accommodate for a wide range of interfaces and levels of abstraction. So, we don't anticipate any significant problems in interfacing with other components of the overarching architecture. In addition, we have already verified that switching to an indoor environment like the one in Figure 12 presents no new difficulties for our architecture.



Figure 12: An agent exploring an indoor environment.

So, here is how we can imagine a possible interface with our unit (Figure 13).

- (i) Our input-output (I/O) module will be linked to the streams of processed sensory input and motor output command, as well as to representations of plans and goals. This I/O connection will allow our unit to follow all major sensory, behavioral and cognitive events that take place in the overarching architecture.
- (ii) In addition, direct links will be made between our schemas (stored in our SM) and symbolic representations in other components. E.g., these links will allow for a synchronous retrieval of appropriate knowledge. Similarly, connections will be made in order to synchronize other components of our unit with the corresponding components outside of it.

What will be the result of this integration? On the one hand, our self-aware unit will reflect on all essential sensory, behavioral and cognitive events that will take place in the greater architecture, with the capability of guiding or vetoing actions. On the other hand, the self-aware unit will be able to handle those situations in test paradigms that may require human-like thinking (self-awareness, meta-cognition, strategic retrieval of episodic memories, cognitive growth, social competency, etc.), when otherwise the architecture could be stuck. We have recently analyzed examples of cognitively challenging paradigms to make sure that we can handle them (e.g., papers by Samsonovich, Ascoli & De Jong in proceedings of IJCNN 2006 and ICDL 2006).

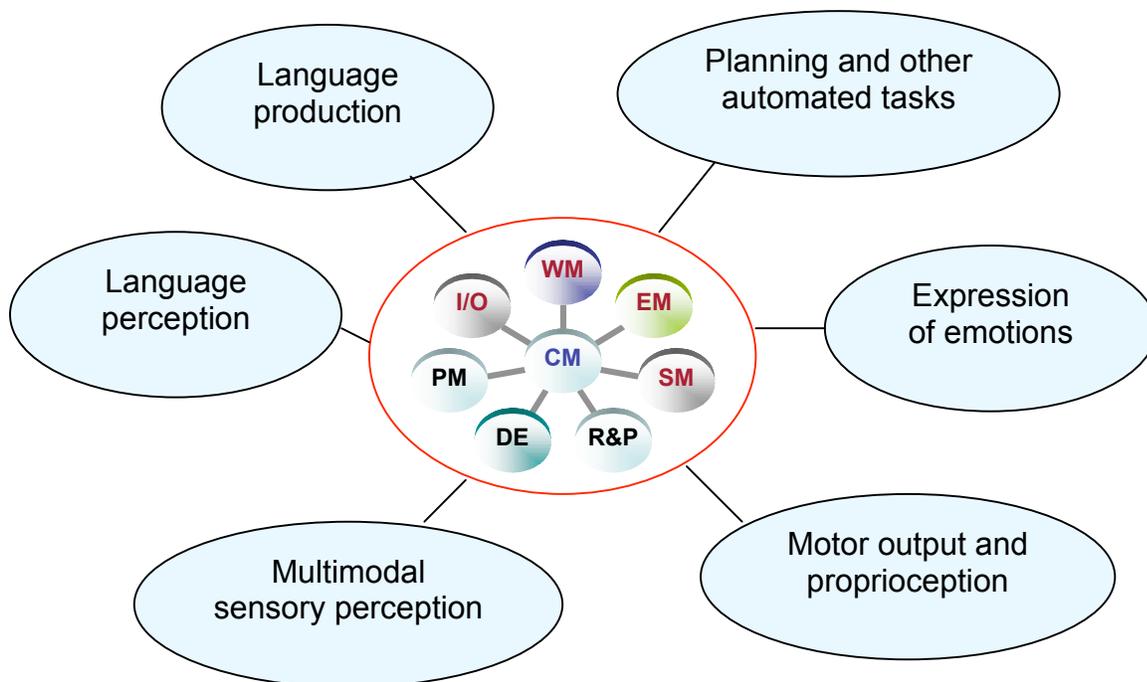


Figure 13. A template for integration.

References

- Albus JS, Meystel AM (2001) *Engineering of Mind: An Introduction to the Science of Intelligent Systems*. New York: Wiley, 2001
- Baron-Cohen S, *Mindblindness: An Essay on Autism and Theory of Mind* (1995) MIT Press, Cambridge, MA.
- Bartsch K, Wellman H (1995) *Children Talk About the Mind*. Oxford UP, New York.
- Carroll JB (1997) Psychometrics, intelligence, and public perception. *Intelligence* 24: 25-52.
- Jensen AR (1998) *The g Factor: the Science of Mental Ability*. Praeger: Westport, Connecticut.
- Goldman A, In defense of the simulation theory, *Mind and Language* 7 (1992) 104-119.
- Mueller ET (2006) *Commonsense Reasoning*. Morgan Kaufmann: San Francisco.
- Nichols S and Stich S, *Mindreading: An Intergrated Account of Pretence, Self-Awareness, and Understanding Other Minds*. (Oxford UP, Oxford, 2003).
- O’Keefe J, Nadel L (1978) *The Hippocampus as a Cognitive Map*. Clarendon: New York, NY.
- Samsonovich A, De Jong KA, Ascoli GA (2006) The integrated self-aware cognitive architecture project. In Hameroff SR. et al. *Toward a Science of Consciousness 2006*, #174. Imprint: Thorverton, UK. Available online at <http://www.consciousness.arizona.edu/abstracts.htm>
- Samsonovich A, McNaughton BL (1997) Path integration and cognitive mapping in a continuous attractor neural network model. *Journal of Neuroscience* 17 (15): 5900-5920.
- Samsonovich AV (2005) Hallucinating objects versus hallucinating subjects. *Behavioral & Brain Sciences* 28 (6): 772-773.
- Samsonovich AV (2006) Biologically inspired cognitive architecture for socially competent agents. In Upal, M. A., and Sun, R. (Eds.). *Cognitive Modeling and Agent-Based Social Simulation: Papers from the AAI Workshop*. AAI Technical Report WS-06-02, pp. 36-48. Menlo Park, CA: AAI Press.
- Samsonovich AV, Ascoli GA (2005) A simple neural network model of the hippocampus suggesting its pathfinding role in episodic memory retrieval. *Learning & Memory* 12 (2): 193-208. Available online via email at <http://krasnow.gmu.edu/L-Neuron/resint.html#publist>
- Samsonovich AV, Ascoli GA (2005) The conscious self: Ontology, epistemology and the mirror quest. In Ascoli, G.A., and Grafman, J. (Eds). *Consciousness, Mind and Brain*. Milan: Masson, 2005.
- Samsonovich AV, Ascoli GA (2005) The conscious self: Ontology, epistemology and the mirror quest. *Cortex* 41 (5): 621-636. Available for free download at <http://www.cortex-online.org>
- Samsonovich AV, Ascoli GA (2006) Self-organizing linguistic cognitive maps as a key to the human value and semantic memory systems. *Society for Neuroscience Abstracts*, CDROM, Prog. No. 263. Washington, DC: Society for Neuroscience.
- Samsonovich AV, Ascoli GA (2006) Cognitive map dimensions of the human value system extracted from the natural language. In: Goertzel B and Wang P, (Eds.). *Frontiers in Artificial Intelligence and Applications*, IOS Press. Forthcoming.
- Samsonovich AV, Ascoli GA, De Jong KA (2006) Computational assessment of the ‘magic’ of human cognition. In Duch W, Kasabov N (Eds.). *Roadmap to Human Level Intelligence*. Singapore. Forthcoming.

- Samsonovich AV, Ascoli GA, DeJong KA (2006) Computational assessment of the ‘magic’ of human cognition. Proceedings of the 2006 International Joint Conference on Neural Networks, 8p. (Paper ID 2035).
- Samsonovich AV, Ascoli GA, DeJong KA (2006) Human-level psychometrics for cognitive architectures. In Smith L, Sporns O, Yu C., Gasser M., Breazeal C., Deak G., Weng J. (Eds.). Proceedings of the Fifth International Conference on Development and Learning ICDL 2006, CD-ROM, 6p. ISBN: 0-9786456-0-X. © Department of Psychological and Brain Sciences, Indiana University: Bloomington, IN.
- Samsonovich AV, Ascoli GA, DeJong KA, Coletti MA (2006) Integrated hybrid cognitive architecture for a virtual roboscout. In Beetz, M., Rajan, K, Thielscher, M., and Rusu, R.B. (Eds.). Cognitive Robotics: Papers from the AAAI Workshop. AAAI Technical Report WS-06-03, pp. 129-134. Menlo Park, CA: AAAI Press.
- Samsonovich AV, DeJong KA (2005) A general-purpose computational model of the conscious mind. In Lovett M, Schunn C, Lebiere C, Munro P. (Eds.). Proceedings of the Sixth International Conference on Cognitive Modeling ICCM-2004, pp. 382-383. Lawrence Erlbaum Assoc., Mahwah, NJ. Available online at <http://simon.lrdc.pitt.edu/~iccm/proceedings/abstracts/Samson.pdf>.
- Samsonovich AV, DeJong KA (2005) Designing a self-aware neuromorphic hybrid. In Thorisson, K.R., Vilhjalmsson, H., and Marsela, S. (Eds.). AAAI-05 Workshop on Modular Construction of Human-Like Intelligence, Pittsburg, PA, July 10. AAAI Technical Report WS-05-08, pp. 71- 78. AAAI Press: Menlo Park, CA. Available online at <http://ai.ru.is/events/2005/AAAI05ModularWorkshop/>.
- Samsonovich AV, DeJong KA (2005) The leverage of a Self concept in incremental learning. In K. Forbus, D. Gentner, & T. Reigier (Eds.). Proceedings of the Twenty-Sixth Annual Conference of the Cognitive Science Society, p. 1627. Lawrence Erlbaum Assoc., Mahwah, NJ. Available online at <http://www.cogsci.rpi.edu/CSJarchive/Proceedings/2004/CogSci04.pdf>
- Samsonovich AV, Nadel L (2005) Fundamental principles and mechanisms of the conscious self. In Ascoli, G.A., and Grafman, J. (Eds). Consciousness, Mind and Brain. Milan: Masson, 2005.
- Samsonovich AV, Nadel L (2005) Fundamental principles and mechanisms of the conscious self. *Cortex* 41 (5): 669-689. Available for free download at <http://www.cortex-online.org>
- Slater M, Usoh M (1994) Depth of presence in virtual environments. *Presence: Teleoperators and Virtual Environments* 3 (2): 130-144.
- Trafton JG, Cassimatis NL, Bugajska MD, Brock DP, Mintz FE, Schultz AC (2005) Enabling effective human-robot interaction using perspective-taking in robots, *IEEE Transactions on Systems Man and Cybernetics: Part A - Systems and Humans* 35: 460-470.
- Tulving E (2004) Episodic memory: From mind to brain, *Revue Neurologique* **160** S9-S23.

Appendix A: An Example of the GMU BICA Design Document

The primary computational specification of the GMU cognitive architecture is the design document that consists of a mixture of text and pseudocode. The final Phase 1 report will include the full design document. For the preliminary Phase 1 report we have included an excerpt that describes one of the 8 top level components, namely the “driving engine”.

Driving Engine Design Document

The driving engine runs the entire system. It is responsible for creating, destroying, and moving most of the system objects. Note that “DE” is a short-hand for “Driving Engine”, “WM” for “Working Memory”, etc. We will visit each Driving Engine method in turn in this section.

DE::start()

DE::start() is invoked when the system is started, or “awakens.”

```
DE::start()
    if birth then:  “i. e., this is the first time started”
        create mental state WM::I-Now
    else if awaken then:  “restoring previous mental states”
        restore mental state WM::I-Now
        restore any other mental states

    mainLoop()
```

DE::mainLoop()

The mainLoop() is the primary Driving Engine function.

```
DE::mainLoop()
    while awake do:
        perceive()
        understand()
        voluntaryAction()
        realityCheck()
        updateMentalStates()

    sleep()
```

DE::perceive()

perceive() is responsible for responding to, or processing new input in the form of instances.

DE::perceive()

```
PM::mapSignalToSymbol() “schema of input from environment appear
in IO”
```

```
set time of WM::I-Now to current system time
```

```
“Now synchronize IO schema with WM schema”
```

```
for @ Schema, SIO, in IO do:
```

```
  if SIO does not exist in WM::I-Now then:
```

```
    clone SIO into WM::I-Now::SI-Now
```

```
    “now update links between SIO and SI-Now”
```

```
    SI-Now.parent = SIO
```

```
    add SI-Now to SIO.clones()
```

```
    (update other attributes for SIO and SI-Now)
```

```
  else “already exists in WM::I-Now”
```

```
    (synchronize existing SI-Now; e.g., update position)
```

DE::understand()

Think about what we already know; note that nested loops will be replaced by more sophisticated mechanism in subsequent example designs.

DE::understand()

```
Sv = [] “empty list of virtual schema instances”
```

```
for @ Schema, SiSM, in SM, do:
```

```
  for @ Schema, SjI-Now, do:
```

```
    Sv-tmp = partialMatch(SiSM, SjI-Now)
```

```
    if Sv-tmp is not null then:
```

```
      append Sv-tmp to Sv
```

```

    “now we have a set of virtual Schemas instances in WM created from
    bind() that will need to be resolved”
    for @ virtual Schema, Sv, do:
        process(Sv)

```

DE::partialMatch()

This function will create any virtual schema instances.

```

DE::partialMatch( SiSM, SjI-Now )
    Sv = [] “empty list of virtual schema instances”

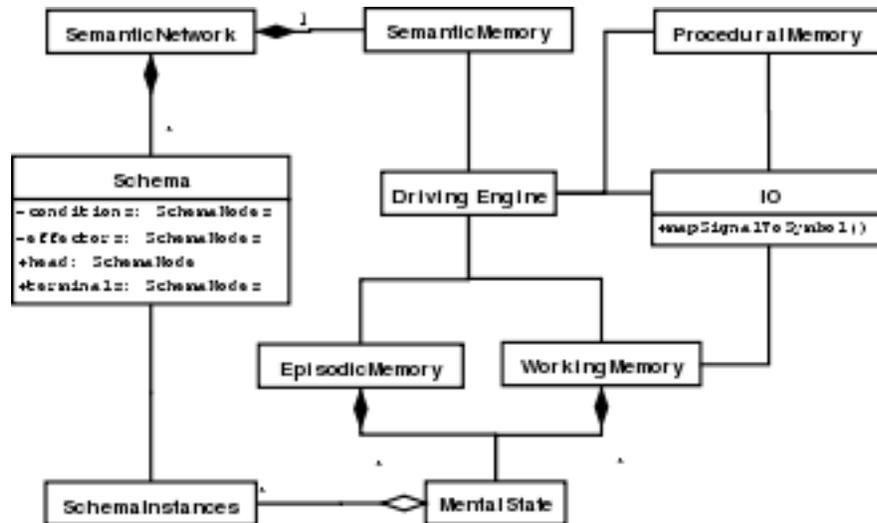
    for @ node, nkSM, in SiSM, do:
        if match( nkSM, SjI-Now.head ) :
            Sv = SjI-Now.clone()
            bind( Sv, SjI-Now.head )
            break “Note that this may mean not all nodes will
                be visited”

    return Sv

```

Appendix B: An Example GMU BICA UML Diagram

Complementing the GMU BICA design document are a set of UML (universal modeling language) diagrams that describe how the various components interact with each other. The final BICA Phase 1 report will include a full set of diagrams for the GMU cognitive architecture. In the preliminary Phase 1 report we have included only an illustrative example.



Appendix C: An Example GMU BICA API

The final piece of the computational specification is self-documenting code. As we implement the prototypes for the key components, we require that the code be self-documenting in the sense that it can be processed by a document generator that produces an API (application program interface) description directly from the code itself. The final Phase 1 report will include a full API description of the implemented prototype. In the Phase 1 preliminary report we have included only an illustrative example.

Module schema

1 Module schema

schema and schema node classes

\$Id: schema.py,v 1.58 2006/09/25 20:06:36 mcoletti Exp \$

1.1 Functions

instanceIn(*i, instances*)

returns true if the instance, *i*, is in the list of instances
XXX maybe make Schema member function?

sameInstance(*i, j*)

returns true if both instances are same
XXX maybe make Schema member function?

1.2 Class Schema

A schema is an idea, meme, or concept chunk. It can be thought of as a template or a class (representing an idea or any other abstraction). Schemas have a list of schemanodes and also can contain within them other schemas. In that sense the definition of schemas is recursive. SchemaNodes are used, among other things to create links between schemas. The first schema node in the terminals list is the 'head' node that contains the Schema specific information.

1.2.1 Methods

__init__(*self, name=None, category=None*)

activation(*self, activation_=None*)

addCondition(*self, condition*)

Add the given condition
XXX should check that this is a Schema type

addEffector(*self, effector*)

Add the given effector
XXX should check that this is a Schema type

addTerminal(*self, terminal*)

Add the given terminal
XXX should check that the terminal is a schema node