The Multipurpose Enhanced Cognitive Architecture (MECA)

Article *in* Biologically Inspired Cognitive Architectures · October 2017

DOI: 10.1016/j.bica.2017.09.006

Ricardo Ribeiro Gudwin
University of Campinas

A.L.O. Paraense
University of Campinas

Suelen M. de Paula
University of Campinas

Eduardo de Moraes Froes
University of Campinas

The Multipurpose Enhanced Cognitive Architecture (MECA)

Ricardo Gudwin[a,*], André Paraense[a], Suelen M. de Paula[a], Eduardo Fróes[a], Wandemberg Gibaut[a], Elisa Castro[a], Vera Figueiredo[a], Klaus Raizer[b]

[a] *University of Campinas (UNICAMP), Campinas, SP, Brazil*
[b] *RLAM Innovation Center, Ericsson Telecomunicações S.A. - Indaiatuba, SP, Brazil*

Abstract

In this paper, we present an introduction to MECA, the Multipurpose Enhanced Cognitive Architecture, a cognitive architecture developed by our research group and implemented in the Java language. MECA was designed based on many ideas coming from Dual Process Theory, Dynamic Subsumption, Conceptual Spaces and Grounded Cognition, and constructed using CST, a toolkit for the construction of cognitive architectures in Java, also developed by our group. Basically MECA promotes an hybridism of SOAR, used to implement rule-based processing and space-state exploration in System 2 modules, with a Dynamic Subsumption Motivational System performing the role of System 1, using a representational system based on conceptual spaces and grounded cognition. We review the conceptual background used on MECA and further provide a detailed description of the many MECA sub-systems.

Keywords: Cognitive Architecture, Dual-process Theory, Dynamic Subsumption, CST

1. Introduction

A cognitive architecture can be viewed as a general-purpose control system inspired by scientific theories developed to explain cognition in animals and men (Langley et al., 2009). The inner behavior of this control system (a mind) is decomposed and explained in terms of a set of cognitive capabilities (Paraense et al., 2016a), as e.g. perception, attention, memory, reasoning, learning, behavior generation, and others. Furthermore, a cognitive architecture can be considered both as a theoretical model, explaining how cognitive processes interact among themselves, and a computational framework, which can be reused throughout different applications. Cognitive architectures have been applied to a broad number of applications such as in robotics, games, human performance modeling, human-machine interaction, natural language processing, virtual agents and others (Kotseruba et al., 2016).

Over the last 20 years, a large set of cognitive architectures has been proposed. In 2010, Samsonovich (2010) created a comparative table, presenting a comprehensive review of implemented cognitive architectures in the literature. Among the highlighted architectures, most of them are general purpose and some of them have their source code available on the Internet to download and to develop simulations. In 2014, Goertzel et al. (2014a) also provided a comparative review of many important cognitive architectures, and now in 2016, Kotseruba et al. (2016) provided a thorough review of 40 years of development of cognitive architectures.

Recently, we started reviewing the many features available in popular cognitive architectures (Lucentini and Gudwin, 2015), and started a project for creating an open-source software toolkit aimed for the construction of general kinds of cognitive architectures (Paraense et al., 2016a). The idea for this toolkit was to be an open repository of cognitive models which might be integrated in generic ways in order to compose specific purpose cognitive architectures. The work being presented in this paper is a first attempt to compose a large generic-purpose cognitive architecture with many features inspired in popular ones like SOAR (Laird, 2012), Clarion (Sun, 2003) and LIDA (Franklin et al., 2014), using our CST toolkit (Paraense et al., 2016a) as a core. We named this cognitive architecture MECA, the Multipurpose Enhanced Cognitive Architecture.

During the design of MECA, we tried to integrate many lessons acquired from our study of other cognitive architectures. Among these lessons, the use of *codelets* (just like in LIDA) as the building blocks of processing, and a mechanism inspired on *Global Workspace Theory* (Baars, 1988) to implement a machine consciousness cognitive capability (similarly but not exactly equals to the one in LIDA), the requirement to have both explicit and implicit knowledge representations (just like in Clarion), considering rule-based processing (just like in SOAR) as an explicit pro-

*Corresponding author
Email addresses: gudwin@dca.fee.unicamp.br (Ricardo Gudwin), paraense@dca.fee.unicamp.br (André Paraense), suelen@dca.fee.unicamp.br (Suelen M. de Paula), edufroes@dca.fee.unicamp.br (Eduardo Fróes), wgibaut@dca.fee.unicamp.br (Wandemberg Gibaut), ecalhau@dca.fee.unicamp.br (Elisa Castro), verafeec@dca.fee.unicamp.br (Vera Figueiredo), klaus.raizer@ericsson.com (Klaus Raizer)

cessing modality, together with a dynamic subsumption-like processing (Brooks, 1986; Nakashima and Noda, 1998; Hamadi et al., 2010; Heckel and Youngblood, 2010) as an implicit processing modality, with the potential to include also neural networks like HTM (George and Hawkins, 2009) within some of its modules. We got inspiration from hybrid cognitive architectures, like SAL (Synthesis of ACT-R and Leabra) (Jilk et al., 2008), which combined both a rule-based architecture (ACT-R) with a neural network one (Leabra) in order to compose a more powerful architecture and mostly a strong inspiration on dual process theory, which recently is being explored under the context of cognitive architectures (Faghihi et al., 2015; Lieto et al., 2016, 2017; Augello et al., 2016). We also relied on important theories regarding knowledge representation, including Grounded Cognition (Barsalou, 2010), Conceptual Spaces (Gärdenfors, 2014) and Computational Semiotics (Gudwin, 2015), extending some recent work (Lieto et al., 2016, 2017) indicating the relevance of extended kinds of representation beyond the traditional symbolic rule-based processing and multi-layered backpropagation neural networks.

The result of this design is presented in the next sections. A set of background theories is reviewed in section 2. Then, in section 3 we present the main MECA specification and finally, in section 4 we provide some conclusions and the next steps of our research.

2. Foundational Background

2.1. Dual Process Theory

The name *Dual Process Theory* relates to a convergent set of theories in Cognitive Psychology about modeling higher cognition within largely disconnected literatures in cognitive and social psychology (Osman, 2004; Evans, 2003, 2008; Evans and Stanovich, 2013; Frankish, 2010; Kahneman, 2011; Osman, 2004; Sloman, 1996; Stanovich and West, 2000). All these theories have in common the distinction between cognitive processes that are fast, automatic and unconscious, and those that are slow, deliberative and conscious. A number of authors have suggested that there may be two architecturally (and evolutionarily) distinct cognitive systems underlying these dual-process accounts. According to Dual Process Theory (Osman, 2004), the mind can be described by the interaction of two different systems, named *System 1* and *System 2*, which assume two functionally distinct roles, which integrate to each other, in order to account for the different facets of the mind phenomena. The exact characteristics of *System 1* and *System 2* varies depending on the theory proposers.

System 1 is generally described as a form of universal cognition shared between humans and animals. It is not actually just a single system, but a a set of sub-systems operating with some kind of autonomy. System 1 includes instinctive behaviors that might be innately programmed and also automatic learned behaviors evolved during the system interaction with its environment. System 1 processes are rapid, parallel and automatic in nature: only their final product is posted in consciousness (Evans, 2003).

System 2 is believed to have evolved much more recently and is considered by many to be uniquely human. System 2 thinking is slow and sequential in nature, and makes use of the central working memory system, intensively studied in psychology. Despite its limited capacity and slower speed of operation, System 2 permits some kinds of abstract hypothetical thinking that cannot be achieved by System 1, as e.g. decision-making using past experiences to abstract new behaviors and the construction of mental models or simulations of future possibilities, in order to predict future events and behave accordingly to reach desirable situations or prescribed goals (Evans, 2003).

Despite their intrinsic autonomy, System 1 and System 2 interact with each other in order to build the overall system behavior. System 1 implements a kind of fast, automatic reactive behavior which provides a default response to system input, aligned with a possible set of general system goals. System 2 has a kind of inhibitory role in suppressing this default response, emphasizing specific time-based goals, which are characteristic of exceptional situations, generating as a result a complex and refined interleaved overall behavior.

There are several important directions for future research. Current theories are framed in general terms and are yet to be developed in terms of their specific computational architecture. An important challenge is to develop models to show how such two distinct systems interact in one brain and to consider specifically how the conflict and competition between the two systems might be resolved in the control of behavior (something we specifically address in our cognitive architecture proposal). According to Evans (2003), theoretical and experimental psychologists need to focus on the interaction of the two systems and the extent to which volitional process in System 2 can be used to inhibit the strong pragmatic tendencies to response in inference and judgment that come from System 1, especially where the latter are known to result in some sort of cognitive bias.

2.2. Subsumption Architecture

The *Subsumption Architecture* is a generic name for a family of computational architectures used in intelligent control (particularly in robotics), developed by Rodney Brooks in the 90's, which gave rise to the whole *Behavior-based Robotics* research field (Arkin, 1998). Brooks proposal was developed under the context of mobile robotics, and was a reaction against the research program on *Artificial Intelligence*, which was suffering from a set of limitations. In the traditional operational cycle of an intelligent control system following the *Artificial Intelligence* paradigm, information was processed by a kind of pipeline, involving perception, internal modeling, planning, task execution and motor control. Even though this operational

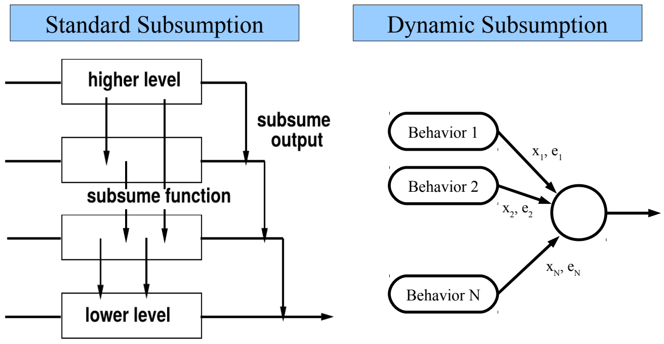

Figure 1: **The Dynamic Subsumption Scheme**

cycle was adequate for small *toy problem* applications, Brooks pointed out the lack of scalability of such architecture. When the number of considered behaviors started to increase, the system complexity started to grow into many problems and the cycle started to become unfeasible.

Alternatively, instead of a serial pipeline, Brooks proposed a parallel strategy, named *Subsumption Architecture*, where new behaviors could be developed in an isolated way and further integrated in the overall architecture. All behaviors used to compete with each other in order to have access to the actuators.

One possible shortcoming to list in Classical Subsumption Architecture is that suppression nodes have fixed dominant and non-dominant nodes. This means that once a behavior is in a higher level, it will always have priority in setting up its behavior. Even though this is desirable in some situations, it is always possible to envision situations in which this priority should be reversed, at least in special occasions. This is not possible in classical Subsumption Architecture.

To deal with this difficulty, some authors (Nakashima and Noda, 1998; Hamadi et al., 2010; Heckel and Youngblood, 2010) proposed a *Dynamic Subsumption* scheme, in which there is no fixed dominant input in a suppression node, but this dominance can be changed dynamically in time, according to specific situations.

There are different implementations of this Dynamical Subsumption idea, but a very simple comprehension of the idea can be taken from figure 1. In the left side of this figure, we can see a standard Subsumption Architecture, with its fixed priority among behaviors. On the right side, we can see how a Dynamical Subsumption can be implemented. Together with the standard control message x_i, there comes together an evaluation tag e_i which is dynamically generated by the own behavior. So, instead of choosing the output value x using a fixed set of priorities, the dynamical mechanism simply chooses the x_i which has the greatest e_i value. This e_i value can be generated dynamically according to the systems' requirements.

2.3. Grounded Cognition

One important theoretic background regarding the issue of *knowledge representation* is the theory on *Grounded Cognition*, provided by Barsalou (1999, 2008, 2010).

Most classic artificial intelligence (AI) studies relied on the concept of *symbol*, using symbolic logic as a background. Propositions, predicates, rules, features lists, frames or semantic networks are typical examples of such structures, most of them defined by an external expert. One of the main criticisms to classic AI was this requirement for a "human in the loop", either acquiring and setting up knowledge, or interpreting the system results. It is not possible to say that these systems (using classic AI) really "understand" the meaning of symbols they use. Barsalou (1999) draws our attention to the fact that these symbols are *amodal* and arbitrary. Their internal structures bear no correspondence with the perceptual states that produced them and therefore they are linked arbitrarily. Thus, *amodal* symbols always require additional representational structures to express meaning. Fortunately, *Computational Intelligence* (CI)[1] came for the rescue, with its many algorithms for classification, categorization, clustering and grouping, based on partial or vague information, and suitable to provide cumulative layers of perception in terms of abstractions of input signals. E.g., neural networks (or fuzzy systems) can be directly connected to sensors and actuators (i.e. connected to the "real world"), and so provide this link to reality which was missing in classic AI. But now we are on the opposite side of the problem. Where are the symbols within a neural network or a genetic algorithm? Are they symbols for the system, or are they symbols for the system designer?

To address this issue, the *Perceptual Symbol System* theory proposed by Barsalou (1999) assumes that the meaning of symbols occurs by the re-enactment of experiences aroused during the acquisition of concepts from the real world.

Barsalou explains that this comprises a new category of symbols, which he calls *perceptual symbols*. In the human mind, *perceptual symbols* might be associated to dynamic neuronal patterns, in structures he names *simulators*. In simulators, information is combined from different sensory sources and aggregated in order to constitute meaning. Thus, two processes are required for the development of a perceptual system: (i) the storage of multi-modal states in order to create simulators (arriving by perception, action and introspection, in long-term memory) and (ii) the partial re-enactment of these states generating a mental simulation.

Differently from amodal symbols, perceptual symbols are analogical and modal, because they are directly represented by the perceptual states which produce them. Consequently, a representational system based on both kinds

[1] In the literature of intelligent systems, the term "Computational Intelligence" is used to designate computational techniques based on Neural Networks, Fuzzy Systems and Evolutionary Computation

of symbols, supports both perception and cognition, without the requirement of a human expert to ground them (Barsalou, 1999). This is specially interesting in language systems since perceptual symbols, accessed during simulation, can represent an object even when its referent does not exist in the physical world, which allows the representation of abstract concepts.

There is somewhat a consensus on the application of perceptual symbol systems theory as an alternative to deal with the symbol grounding problem in computer simulations. This can be viewed by the increasing number of works using this approach to support their models elaboration (e.g. Narayanan (1999), Bergen et al. (2004), Roy (2005), Cangelosi and Riga (2006), Dominey et al. (2006),Madden et al. (2010), Lallee et al. (2010), Frank and Vigliocco (2011) and Pezzulo and Calvi (2011)).

2.4. Conceptual Spaces

A second theory which is also related to knowledge representation is the theory of *Conceptual Spaces* from Gärdenfors (2004, 2014). This semantic theory considers that the mind organizes information involved in perception, attention, categorization and memory, using geometric and topological properties, in order to derive the notion of conceptual spaces.

Conceptual spaces are metric spaces providing a robust mechanism for learning and representing the meaning of different classes of words (e.g. categories of objects). According to Gärdenfors (2014), an unified theory of meaning about different word classes can be developed when conceptual spaces are used to provide linguistic grounding. In our conception, this theory is complementary to Barsalou's theory by exploring both semantic and lexical aspects of language. Besides, what Barsalou has defined as perceptual symbol, Gärdenfors defines as object categories and both are special kinds of a concept. Moreover, Gärdenfors concepts may involve perception but also memory, attention and imagination, while concepts from Barsalou's theory are formed only by perceptual experiences.

According to Gärdenfors (2014), *concepts* are mathematical structures which fully represent the meaning of words. The most common kind of concepts are object categories, but there might be concepts associated to qualities, actions, events and possibly to all categories of words and special combinations of words as well. Concepts are defined with the help of *conceptual spaces*, and conceptual spaces are constructed out of *quality dimensions*. The primary role of these dimensions is to represent various "qualities" of objects in different *domains*.

A *domain* corresponds to a set of integral dimensions that are separable from all other dimensions. Many domains, such as *temperature* or *weight*, consists of only one dimension. Other domains, such as *color*, or *location*, might require multiple dimensions. A conceptual space is a collection of one or more *domains*, which can be used to assign *properties* to an object. Therefore, objects are identified as points within conceptual spaces. Their properties are represented by regions in specific *domains* and the *category* of an object, which is also a concept, is denoted by a collection of regions (properties) and their relations in a conceptual space.

To have a space partitioned into a finite number of regions means that a finite number of words can be used to refer to such regions. Therefore, conceptual spaces provide a robust framework for learning concepts for language.

According to Gärdenfors (2004), the use of conceptual spaces provides a different approach when compared to symbolic AI. In symbolic AI, the assignment of semantics to symbols requires an external interpretation. In the current approach, the semantics is implicit in the definition of conceptual spaces. Therefore, the use of conceptual spaces might be a possible aid for the solution of the symbol grounding problem.

2.5. CST: The Cognitive Systems Toolkit

To finalize our Theoretical Background section, it is important to mention CST - The Cognitive Systems Toolkit (Paraense et al., 2016a), which is used as the basic infrastructure for the construction of MECA.

The most essential concepts in CST's core are the notions of *Codelets* and *Memory Objects*. Codelets are small pieces of non-blocking code, each of them executing a well defined and simple task. The idea of a codelet is of a piece of code which ideally shall be executed continuously and cyclically, time after time, being responsible for the behavior of a system's independent component running in parallel. The notion of codelet was first introduced by Hofstadter and Mitchell (1994) and further enhanced by Franklin et al. (1998) and used widely within LIDA architecture. Any codelet-oriented architecture is intrinsically a fully parallel asynchronous multi-agent system, where each agent is modeled by a codelet. CST's codelets are implemented much in the same manner as in the LIDA cognitive architecture (Franklin et al., 2014) and largely correspond to the special-purpose processors described in Baars' Global Workspace Theory (Baars, 1988; Baars and Franklin, 2007).

The second element which is vital to understanding CST's Core is the notion of a *Memory Object*. A *Memory Object* is a generic information holder, acting as a *sign* or an internal representation, which is responsible to store any auxiliary or perennial information necessary for the cognitive architecture to perform its behavior. Codelets and Memory Objects are intrinsically coupled to each other, in the sense that Memory Objects hold the information necessary for the Codelets to run, and are also the placeholders of any new information generated by the codelet. The main property being hold by a Memory Object is its Information (I). This information can be simply a number, or hold complex structures like lists, trees, graphs or whole networks. In our computational implementation, the information I is a generic Java Object, which can be used to represent virtually anything[2]. A Memory Object also has two extra kinds of meta-information: a time stamp (T), which tags the Memory Object with a marker indicating its last update, and an evaluation (E), which has many different uses within CST. This evaluation is simply a number, which can be used, e.g. as an appraisal factor in an emotional module, or simply as a discriminator to differentiate among two or more Memory Objects of the same type. These meta-information can be simply ignored by simpler codelets, or be useful while implementing more sophisticated cognitive models.

Together with the introduction of CST, Paraense et al. (2016a) also describe the CST Reference Cognitive Architecture, a reference guide for how to create a cognitive architecture with CST. They call it a Reference Architecture because it is an abstract view of how to organize a set of codelets and a set of memory objects, in order to build a cognitive architecture. Codelets are organized into groups, each group responsible for implementing a cognitive model of some cognitive function, Like Sensing, Perception, Attention, Emotions, Learning, Language, Consciousness, Imagination, Planning, Behaviors and Motor Actuation. The CST toolkit provides standard implementations for some of these groups. Other codelets might be available in the future, as CST implementation evolves. New types of codelets can be created by the cognitive architect, and easily bound together using CST core functions.

The homepage of the CST Project can be found at `http://cst.fee.unicamp.br`, and its source code is available as open-source in GitHub at `https://github.com/CST-Group/cst`.

3. The MECA Cognitive Architecture

The conception of MECA inherits a lot of insights coming from different sources. Most of them were previously presented in section 2. First of all, MECA is an instance of the *Dual Process Theory*. This means that MECA is split into two major sub-systems, *System 1* and *System 2*.

We start our description of MECA by introducing an overview of the whole architecture, describing its main components. After that, we start detailing each of these components.

3.1. An Overview of the Whole Architecture

An overview of the MECA Cognitive Architecture can be seen in figure 2. The architecture is specified following the CST Reference Architecture (see (Paraense et al., 2016a)).

The whole architecture is split into two major subsystems, as indicated in figure 2: *System 1* and *System 2*. These two sub-systems exchange some information, such that they interact with each other in order to generate its overall behavior.

The architecture is designed as a network connecting 4 kinds of elements: codelets, memory objects, containers (a special kind of memory object) and memories, as indicated in figure 3.

System 1 is comprised by three different memories: *Sensory Memory*, *Perception Memory* and *Motor Memory*. There are many subsystems accessing those memories, which are indicated by codelets of different kinds:

- Sensory Subsystem: Sensory Codelets
- Motor and Behavior System: Behavioral Codelets and Motor Codelets
- System 1 Motivational Subsystem: Motivational Codelets, Mood Codelets, Emotional Codelets
- Perceptual Subsystem: Perceptual Codelets, Attention Codelets

System 2 comprises basically 5 different memories: Perceptual Buffer, Episodic Buffer, Episodic Memory, Working Memory and Procedural Memory. Also, there are many subsystems accessing those memories, implemented by different kinds of codelets:

- Episodic Subsystem: Attention Codelets, Episodic Learning Codelet and Episodic Retrieval Codelet
- Planning Subsystem: Soar Codelet
- System 2 Motivational Subsystem: Goal Codelet and Appraisal Codelet
- Consciousness Subsystem: Consciousness Codelet
- Expectation Subsystem: Expectation Codelet

All architecture inputs and outputs are performed exclusively by *System 1*. The inputs to MECA are performed by the *Sensory Codelets* (on the left of System 1 at the diagram), which are responsible for collecting the sensor data and populating the *Sensory Memory*. The MECA outputs are performed by the *Motor Codelets* (on the right of System 1 at the diagram), which basically collect data from the *Motor Memory* and are responsible for sending this data to the system actuators.

3.2. The System1 Specification

To understand the role of *System 1* in the overall architecture, it is important to recapitulate the role of *System 1* in *Dual Process Theory*. According to the theory, *System 1* is responsible for the unconscious fast automatic behavior

[2] The choice of using a generic Java Object as a general knowledge representation entity was a deliberate choice in order to bring flexibility to the toolkit user. The toolkit might provide specific knowledge representation options, like propositions, predicates, rules, together with more network oriented kinds of representations like in neural networks, Bayesian networks or whole graphs like LIDA's Behavioral Network, SOAR's WME's (Working Memory Elements), or OpenCog atoms and atom spaces (Goertzel et al., 2014b). In order to get modules developed in different contexts to work together, solutions like JSON strings are also an option.

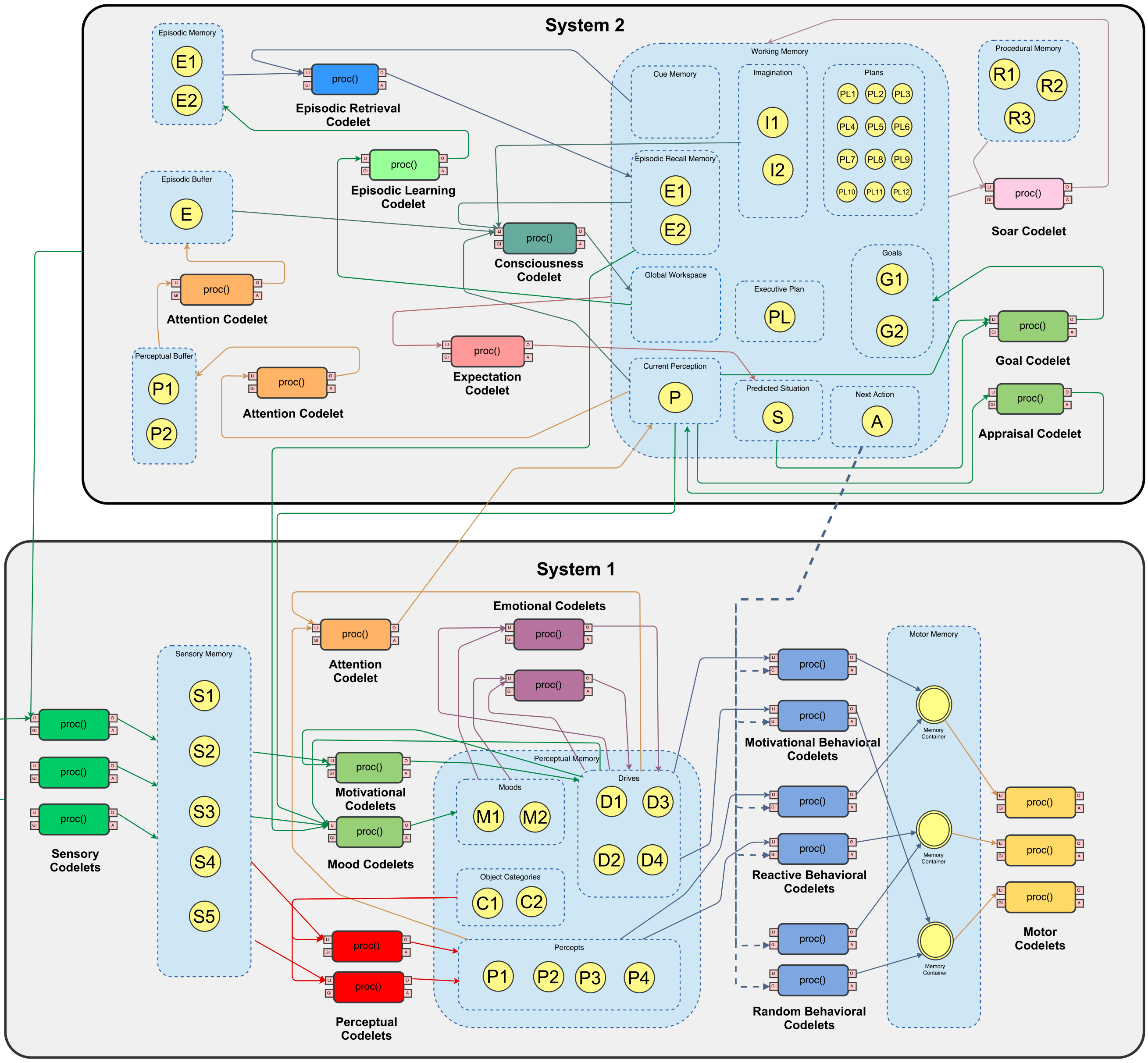

Figure 2: **An Overview of the MECA Cognitive Architecture**

performed by the whole architecture. It is mainly a reactive behavior, even though motivational based behavior (instinct) is also possible. The processing in *System 1* is mostly parallel, and the knowledge is usually implicit.

To implement *System 1*, we designed a Dynamic Subsumption Architecture, implemented on top of CST. The inputs to this Dynamic Subsumption Architecture might come directly from Sensory Memory, but usually there is some kind of Perception processing in between. The role of Perception is to generate more elaborate Percepts, abstractions of sensory data, which are then used as input to the Behavioral Codelets. These percepts can also be tracked by Attention Codelets in order to detect special situations and send information upstream to *System 2*. These attention codelets are responsible for generating the *Current Perception* at the Working Memory, where a selected subset of the Perception Memory is made available for *System 2* subsystems in a representation suitable to be processed within *System 2*.

Among the behavioral codelets, there is a special sub-set (Motivational Behavioral Codelets) comprising the System 1 Motivational Subsystem, which is responsible for implementing a kind of instinct mechanism in the architecture. This Motivational Subsystem also includes some sort of emotional processing.

The details of *System 1* are presented in the next subsections.

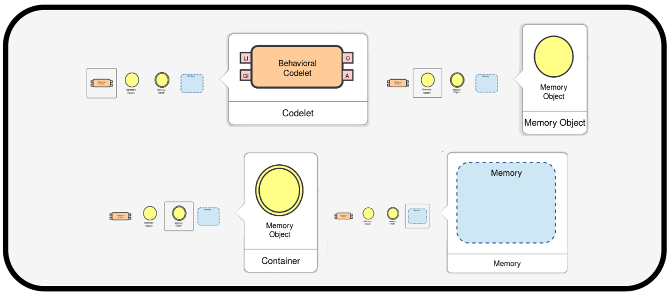

Figure 3: **The Different Elements in the Architecture**

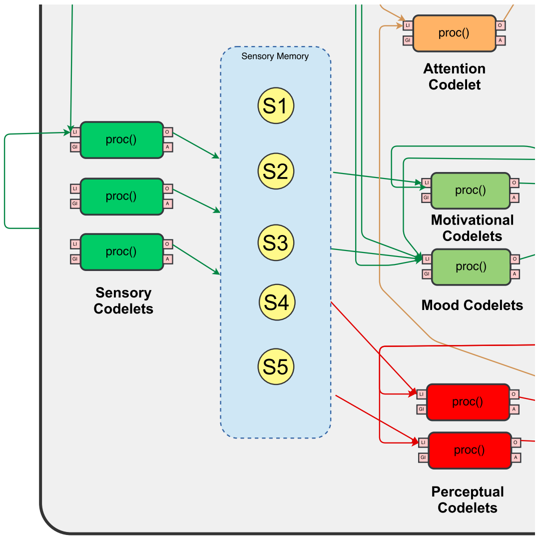

Figure 4: **A Zoom on the Sensory Subsystem**

3.2.1. The Sensory Subsystem

From a cognitive modeling perspective, the *Sensory Memory* is the raw storage of information coming from visual, auditory, olfactory, tactile and other sensory modalities in a time window which generally spans over something as 50-500 ms (Baddeley, 1997). This memory usually comprises uninterpreted data which is used in the first steps of perception. In human beings, at least two kinds of sensory memory were identified, the *Iconic Memory*, storing visual pattern stimulus and the *Echoic Memory*, storing auditory stimulus, but there might be also other memories for other senses as well, not so widely investigated.

In MECA, this memory stores Memory Objects carrying direct representations of system sensors. These Memory Objects can be simple numbers or very complex data structures, representing both scalar sensors or n-dimensional images, according to the information provided by the sensor. It might also store temporal sequences of sensor data, which can be used by Perceptual Codelets to create more elaborate percepts. More elaborated or derived representations from direct sensory capture are not stored here, but at the Perceptual Memory. The Memory Objects stored in the Sensory Memory are updated by Sensory Codelets.

Sensory codelets are codelets which are responsible for grabbing information from sensors at the environment, and feeding the corresponding Memory Objects which might hold the sensors values. Depending on the applications (e.g. robotic applications), sensory codelets will be really reading the sensor values and creating a corresponding representation. In other applications (as e.g. in a video-game, an internet agent or a virtual world), sensory codelets will open sockets to other computer applications and will simulate the acquisition of data from the environment.

Usually, Sensory Codelets are application specifics, and the MECA software implementation just provides basic template classes to be reused while building an application using MECA.

3.2.2. The Behavioral and Motor Subsystem

The Behavioral and Motor Subsystem is the core of *System 1*, and is focused in figure 5.

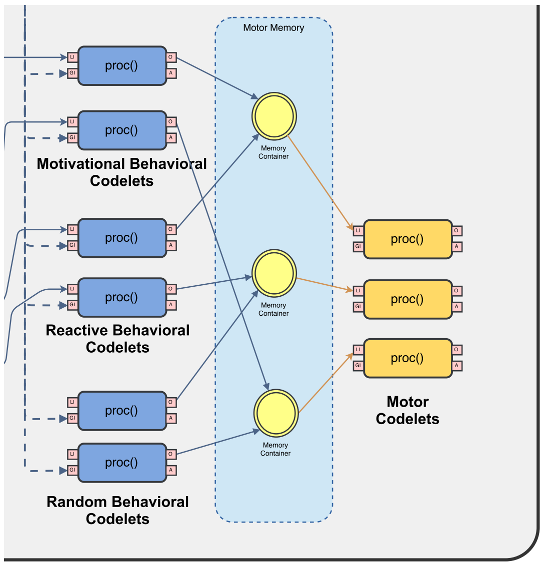

Figure 5: **A Zoom on the Behavioral and Motor Subsystem**

The *Motor Memory* is a direct representation of the system's actuators. Memory Objects in the Motor Memory are usually actuator values, which will be used as parameters by Motor Codelets in order to actuate at the environment. But the Motor Memory, in MECA has a special kind of implementation. Instead of using simple Memory Objects, MECA uses a special kind of Memory Object which is called a *Container* (see figure 6).

The Container Memory Object is responsible for implementing an important element in the Dynamic Subsumption mechanism used in MECA. Take the example of figure 5. The first Container in Motor Memory is receiving an input from a Motivational Behavioral Codelet and another from a Reactive Behavioral Codelet. Both codelets are proposing a behavior which uses the same actuator. The system needs to decide which actuator parameters it will accept in order to send to the real actuators. This is all performed internally within the Container. All the

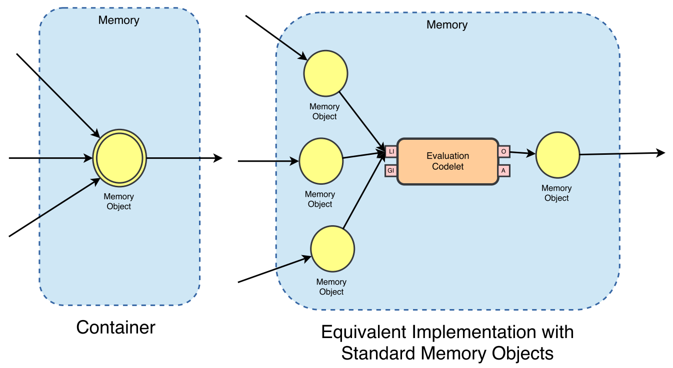

Figure 6: **A Container Memory Object and its Implementation**

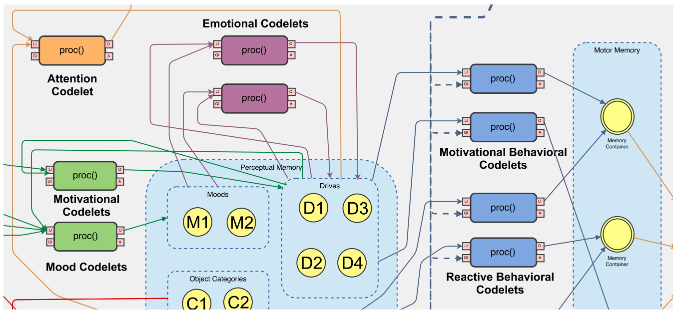

Figure 7: **A Zoom on the System 1 Motivational Subsystem**

Memory Objects in a Container are of the same type, and hold the same parameters. The only differences among them are that they were generated by a different behavior codelet, and they might have different *evaluations*. An *evaluation* is an inner parameter from any Memory Object, which holds a value (usually a real value between 0 and 1) that measures a relative importance given by the behavioral codelet, and which is used by the Evaluation codelet within the Container to decide which from all input Memory Objects will be sent to the Motor Codelets. The behavior of the Evaluation Codelet is very simple. It simply reads all the *evaluation* parameters from all input Memory Objects, and select the Memory Object with the highest *evaluation* to send to output. To simplify the diagrams, we chose to represent this scheme by a specific icon in our diagrams, because in fact the Container works just like a standard Memory Object. The outputs from different behavioral codelets are merged into just one Memory Object which is redirected to its correspondent Motor Codelet.

Motor codelets then simply pick up the result Memory Object from the Motor Memory and reacts directly at the environment. This can be done by simply capturing actuator values and feeding actuators, or by some special protocol interacting with external software or hardware.

3.2.3. The System 1 Motivational Subsystem

The *System 1* Motivational Subsystem is depicted in figure 7. Motivational behavior in cognitive architectures is derived from studies on human motivation coming from psychology, like those from Maslow (1943) and Hull (1943). Examples of sophisticated models which served as inspiration to our Motivational Subsystem are those from Sun (2009); Sun and Wilson (2011); McCall (2014).

In order to properly understand what is a motivational behavior (or a *motivated* behavior), it is important to clearly distinguish it from random behaviors and purely reactive behaviors. A random behavior does not depend on anything else. It is, as Peirce proposes in Semiotics theory, a *firstness*, in the sense that it is completely independent of anything else. A purely reactive behavior is a behavior which is simply triggered by sensory input. This means that every time some input data is sensed, this will trigger this behavior. A purely reactive behavior is what Peirce calls, in Semiotics theory, a *secondness*, in being something which depends on something else. A motivated behavior, instead, do not depend on just the present input, but depends also on a future state the system is aimed to reach. In other words, the behavior is not simply a reaction to some input, but is a behavior which is oriented towards a future state. It depends both on current input and on this desired future it is meant to achieve. This is what Peirce calls a *thirdness* in Semiotics theory. The philosopher Aristotle, in his Theory of Causality, used to call it a *final cause*. Many authors call this kind of behavior a *goal-directed behavior*. It is important to notice that a *goal-directed behavior* is fundamentally different from a reactive or a random behavior. During very much time, final cause and related concepts (e.g. the concepts of purpose or teleology) were viewed with skepticism in science. Modern cybernetics clearly explained though, the mechanism behind goal-directed behavior: feedback loops (Rosenblueth et al., 1943).

According to Hull's theory of behavior (Hull, 1943), when a motor action is a pre-requisite to optimum probability of survival of either an individual or a species, a state of *need* is said to exist. This need is said to motivate or *drive* the associated motor action. So, Hull defines a *Drive* as being an intervening variable used to characterize a need. Drives are used as a measurement of a desirable future state which a creature must reach, in order to survive. In a biological creature, a drive might be related to the many needs a biologic being is supposed to have: need for food, for water, for air, the need to avoid injury, to maintain an optimal temperature, the need to rest, to sleep, to mate, etc. In an artificial agent, *drives* are associated to the desirable behaviors we want the agent to manifest. They have to do with the desirable future state the agent should move itself into. In a very abstract understanding, a *drive* is a measurement of the agent's success in achieving its designed purpose. A behavior which is performed in order to satisfy a drive is said to be a motivational (or motivated) behavior.

A system might have many different drives, each of them encoding some internal purpose designed for the system. This leads to the necessity of creating a whole Motivational Subsystem.

In MECA, Motivational Behavior is split among *System 1* and *System 2*. *System 1* Motivational Behavior is related to long-term or perennial needs for the agent the cognitive architecture is controlling (e.g. to optimize energy costs, to maintain energy balance, etc). They are usually related to default system behavior which is expected from the agent. *System 2* Motivational Behavior is related to time-specific needs, that once satisfied results in the dismission of the need (e.g. to reach a particular state, to achieve a specific goal). They are usually related to an exceptional situation which is under consideration. Needs on *System 1* are encoded as *Drives*. Needs on *System 2* are encoded as *Goals*.

The standard dataflow for the *System 1* Motivational Subsystem starts from Sensory Memory Objects flowing through Motivational Codelets to generate Drives which are stored in the Perceptual Memory. These Drives are then used by Motivational Behavioral Codelets in order to contribute with behaviors to be selected in the Dynamical Subsumption scheme.

But MECA also previews an enhancement in this standard dataflow, which is the incorporation of an emotional mechanism.

The concept of emotion, as brought from cognitive psychology and philosophy, was suggested in the literature, as an alternative way of dealing with the problem of behavior generation (Bates et al., 1994; Reilly, 1996; Picard, 1997; Canamero, 1997, 1998; Septseault and Nédélec, 2005; Budakova and Dakovski, 2006; Meyer, 2006).

In MECA, we will be following Cañamero's (Canamero, 1997) approach to emotions, together with Sun's proposal to a motivational system. Under this view, emotions work as temporary cognitive distortions on system drives, resulting in a change in priorities, due to the recognition of critical situations. These critical situations will be recognized by *Mood Codelets* from direct sensor data, but also from situations remembered from episodic memory and possible predicted situations (from *System 2*) which might be classified as critical. The detection of a critical situation will change the *Moods* in Perception Memory. These Moods are responsible for, through Emotional Codelets, change the Drives intensity landscape, resulting in a change of priorities in order to better attend the critical situation.

3.2.4. The Perceptual Subsystem

The *Perceptual Subsystem* is the subsystem responsible for abstracting the information coming from Sensory Memory and building more sophisticated representations for what is going on at the environment. There might be increasing layers of abstraction in this process, under which raw data measurements are transformed in a high level understanding of the environment situation. This process

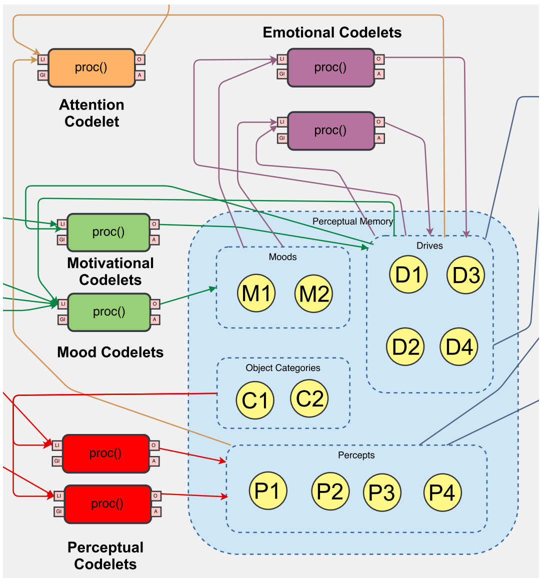

Figure 8: **A Zoom on the Perceptual Subsystem**

is similar to the many layers of an onion, where high-level abstractions sum up and superpose the lower-levels, using their information and composing more sophisticated representations of the environment change over time. This is the process where sensory measured *properties* give rise to derived properties, which are then integrated into the recognition of *objects*, which are then identified as being a part of an ongoing *episode*, constituting a *scene*, which sequence form an understanding of the many situations *experienced* by the agent along the passage of time.

In MECA, the Perceptual Subsystem is integrated with the Motivational Behavior System through the *Perceptual Memory*.

The *Perceptual Memory* is a memory storing the many structures required both for the Perceptual System and for the Motivational Behavior Subsystem. It includes a submemory of *Percepts*, *Object Categories*, *Moods* and *Drives*.

The sub-memory of *Percepts* comprises Memory Objects encoding abstractions or high level representation of objective items of reality. These are usually called in cognitive science as *Percepts*. There might be many different possible ways for representing percepts, like *fuzzy sets*, patterns, objects and other more elaborate representations.

Perceptual Memory is mainly fed by Perceptual Codelets, which collect information from Sensory Memory Objects and, using the object categories in the *Object Categories* sub-memory, provide high-level abstractions of sensory data in terms of percepts. Also, Motivational Codelets feed Moods and Drives at the Perceptual Memory. Many categories of codelets may use Perceptual Memory Objects as a source of information.

The Perception Subsystem is also responsible for collecting percepts from the Perceptual Memory and sending this information to *System 2*. This is done by an Attention codelet, which picks the current list of percepts and creates a *Current Perception* representation at the Working Memory. According to the CST Reference Architecture, Attention codelets are specialized kinds of codelets which will work as salience detectors for objects, situations, events or episodes happening at the environment which might be important for defining an action strategy, or behavior. In the case of MECA Perception Subsystem, an attention codelet is responsible for analyzing the many percepts at the Perceptual Memory and deciding which of them are important enough to be a part of the *Current Perception* at the Working Memory in *System 2*. This process is an important part of the consciousness mechanism in System 2, and also important for the Episodic Subsystem, also in System 2.

3.3. The System2 Specification

According to *Dual Process Theory*, *System 2* is responsible for the slow conscious process of deliberative reasoning. It is mainly a sequential rule-based process, operating on symbols, and considering not just the present, like in *System 1*, but also the past and the future. This is the place where imagination and planning occurs. This is also the place where the many unconscious perceptions performed at *System 1* enter into a process of competition to integrate the agent's present *experience*, where the most important percepts are payed attention and other less relevant are discarded. This leads to the formation of the conscious perception which is usually called *experience* by many philosophers of mind, and which are integrated into *episodes*, and then stored in an episodic memory to be recovered later, for many purposes.

The *System 2* Specification in MECA includes the definition of an Episodic Subsystem, responsible for higher-level perception with the tracking of time along Perceptual Memory and with the aid of Attention codelets discover and detect the formation of *episodes*, and the storage and recovering of these episodes in the Episodic Memory. It also includes a Planning Subsystem, responsible for simulating the future and making plans of action in order to reach possible *Goals*. The Planning Subsystem is also responsible for the process of Imagination, which is used as an aid for testing possible courses of action and evaluating the best action to take. MECA's implementation of *System 2* also includes a High-Level Motivational Subsystem, responsible for generating *Goals* for the Planning Subsystem, an Expectation Subsystem which tries to foresee the short-term future and learn from the possible inconsistencies found, and a Consciousness Subsystem, responsible for filtering the information available for the Planning Subsystem.

The details of the *System 2* are presented in the next subsections.

3.3.1. The Episodic Subsystem

MECA's Episodic Subsystem is illustrated in figure 9. The Episodic Subsystem in MECA has basically two roles:

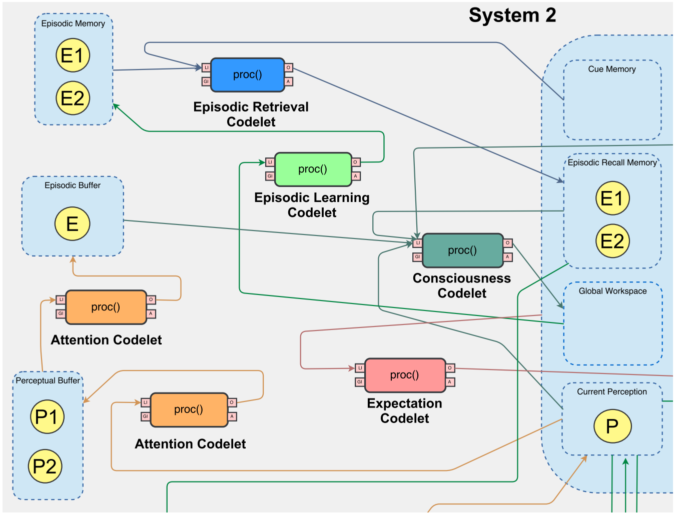

Figure 9: **A Zoom on the Episodic Subsystem**

- To implement a higher-level kind of perception, and based on sequences of collections of percepts in time (which in MECA terminology are called **configurations**), detect and represent *Episodes* describing scenes experienced by the MECA agent.
- To store and retrieve these episodes in/from *Episodic Memory*

In order to perform the first of these roles, the Episodic Subsystem uses two auxiliary memories: the *Perception Buffer* and the *Episodic Buffer*, and two Attention Codelets. Everything starts with the *Current Perception* at the Working Memory. This *Current Perception* was set up by the Perceptual Subsystem in System 1, and is basically a representation of a *Configuration*, a collection of percepts from *Perceptual Memory* which was diagnosed as important or relevant enough to be sent to *System 2*.

According to the theory on Cognitive Modeling, the *Episodic Buffer* is a limited capacity storage buffer which binds together information from a number of different sources into chunks or episodes, combining information from different modalities into a single multi-faceted code in order to be processed by the Central Executive (Baddeley, 2000). The *Episodic Buffer* comprises the detection of episodes at the environment, as they are happening. The information on the Episodic Buffer is an abstract representation of the perceived present. The structures in the Episodic Buffer are the episodes which will later be stored in the Episodic Memory, in a sequence, forming a continuous timeline where we can recover episodes from the past.

In standard Cognitive Modeling theory, the *Episodic Buffer* is also a part of Working Memory. In current MECA implementation, we decided to isolate the Episodic Buffer out of the Working Memory. This is basically a development strategy, because this process of recognizing and representing an episode is a very complex one and we would like not to deal with potential complications if the Episodic Buffer was a part of Working Memory. This might change in future implementations of MECA, as there is evidence from the literature that the *Episodic Buffer* is really a part of Working Memory.

As we said, the process of detecting and recognizing an episode is a very sophisticated one. In order to explain our MECA implementation of this process, we need to first explain some issues related to the scientific discoveries regarding *Episodic Memory*.

The *Episodic Memory* (which should not be confused with the Episodic Buffer) is a memory used to store facts particularly contextualized in time and space, forming *Episodes* which refer to information specific to a particular location and time frame. Episodes are representations for scenes detected from environment, using a higher level abstraction of space-time. We can see an episode as a specific representation for a segment of space-time, where some specific set of objects, and their trajectory in their state space is somewhat represented.

Episodic Memory is a neurocognitive mechanism for accessing time delimited contextualized information that naturally makes part of the human process of decision making, usually enhancing the chances of a successful behavior. This assertion is supported by several human psychological researches which indicate that the knowledge of his/her personal history enhances one's person ability to accomplish several cognitive capabilities in the context of sensing, reasoning and learning (Tulving, 1991; Baddeley, 2000, 2002; Tulving, 2002; Howard et al., 2005; Cabeza et al., 2008).

Our decision was to implement both *Scene-based episodes* and *State-based episodes* in MECA, something very different from other known Cognitive Architectures available in the literature. The reason for that is due to our commitment with Language Processes to be developed within the scope of MECA. The structure of *Grammar Language*, i.e. the use of sentences with meaning and understanding in artificial systems is directly connected with the interpretation of scene-based episodes.

The construction of episodes within the *Episodic Buffer* required us an additional structure. For that sake, we introduced the *Perceptual Buffer*. The whole idea of detecting an episode follows the further sequence of events.

At each time step, the Current Perception in Working Memory holds a Configuration, including all relevant discovered objects at the environment, and its parameter values. The first attention codelet in the Episodic Subsystem then compares this *Configuration* with a sequence of other *Configurations* stored in the *Perceptual Buffer*. If it is different enough from the last *Configuration* stored, it then decides to include it in the Perceptual Buffer. The *Perceptual Buffer* works as a FIFO (First-In First-Out) list. It stores a sequence of the last detected *Configurations*, as they appeared at *System 2*. The second Attention codelet then scans the list of *Configurations* in the *Perceptual Buffer*, trying to detect the beginning and the end of an *Episode*, mounting it in the *Episodic Buffer*. As a full *Episode* is detected and released in the *Episodic Buffer*, this episode becomes available to the Consciousness Mechanism (see section 3.3.4) to gain access to the Global Workspace. Whatever is in the Global Workspace is then collected by the *Episodic Learning Codelet* and stored in the *Episodic Memory*. Finally, as soon as many *Episodes* are already stored in the *Episodic Memory*, the *Episodic Retrieval Codelet* can perform its abilities. The behavior of the *Episodic Retrieval Codelet* is quite straightforward. Basically, it collects a cue from the *Cue Memory* (which is basically populated during the working of the *Planning Subsystem*, tries to recover pertinent episodes from the *Episodic Memory* and brings these episodes to the *Episodic Recall Memory* within the Working Memory, from where they become available for the Planning Subsystem.

3.3.2. The Planning Subsystem

An overview of the *Planning Subsystem* can be viewed in figure 10;

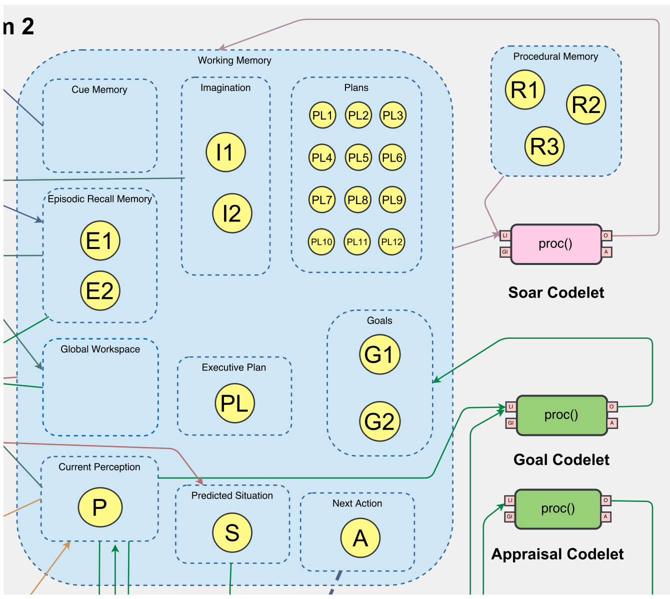

Figure 10: **A Zoom on the Planning Subsystem**

The *Planning Subsystem* is the core of *System 2*. The current implementation of the *Planning Subsystem* relies on SOAR, a rule-based Cognitive Architecture developed by John Laird at the University of Michigan, in USA (Laird, 2012). This can be changed in a future release of MECA. The decision to use SOAR for rule-based processing in order to perform imagination and planning was due to our time constraints in the development of MECA, and depending on the further results might change to a different solution. This decision was aligned with recent tendencies in the literature. The cognitive architecture SAL

(Synthesis of ACT-R and Leabra) performs exactly something like that, binding two different cognitive architectures (ACT-R and Leabra), where ACT-R is a rule-based cognitive architecture, similar to SOAR, and Leabra is a Neural-network based cognitive architecture. The choice of SOAR instead ACT-R was due to the fact that there is an implementation of SOAR in Java (ACT-R is in LISP), and our CST toolkit is also in Java, allowing for a better binding and use of SOAR instead ACT-R. Besides this technological issue, ACT-R or any other rule-based cognitive architecture might also be used instead of SOAR. SOAR is though a mature technology, with a very good documentation and a solid repertoire of use in the community, and so it appeared to be the right choice.

The role of the *Planning Subsystem* is to plan a course of actions that, starting with the current situation, detected by *System 1*, find the sequence of operations required to reach a target situation specified by a *Goal*. This process, in current MECA's implementation, is executed by SOAR, and is equivalent to the notion of a *mental simulator* proposed by Barsalou (1999) in *Grounded Cognition*.

Behind the scenes in the *Planning Subsystem*, there is the structure of *Working Memory*.

According to theories from cognitive modeling, the *Working Memory* is a volatile kind of memory used during perception, reasoning, planning and other cognitive functions. In studies with human beings, its capacity in time and space is found to be very short, ranging from 4 to 9 items, and periods up to a few dozen seconds (Miller, 1956; Baddeley et al., 1975; Cowan, 2001). According to Baddeley (1997, 2000), there are at least three subsystems involved in the implementation of a Working Memory, the *Visuo-spatial Sketchpad*, the *Phonological Loop* and the *Episodic Buffer*, coordinated by a *Central Executive* which intermediates between them. Regarding brain localization, the regions related to working memory processes are very overlapping, however recent researches point the prefrontal cortex and basal ganglia as being crucial (Braver et al., 1997; Frank et al., 2001; McNab and Klingberg, 2008).

In the current MECA implementation, we are not relying in this structure. The *Episodic Buffer* is split from the Working Memory and we don't have neither the *Visuo-spatial Sketchpad* neither the *Phonologic Loop*. In future implementations of MECA, this might change, though. In its current implementation, the Working Memory is a repository of *Symbols*, which are grounded on *Percepts* from the Perceptual Memory, and which are used to plan the future, through a process of *Computational Imagination*.

Computational Imagination (Setchi et al., 2007) is a cognitive function described in many works (Chella et al., 2005; Marques and Holland, 2009) as being the ability to simulate and test different possible scenarios, allowing the construction of plans. In this sense, imagination and planning are bounded together. In the current implementation of MECA, the process of *Computational Imagination* is performed by SOAR, with the help of Episodic Memory, but other approaches for planning, including Prolog, ACT-R or others are possible. In fact, any kind of rule-based system available in Java can be linked to CST and be a part of MECA.

The *Procedural Memory* is the memory of actions and behaviors of a system. According to cognitive modeling theory, it is a non-declarative memory which refers to a "how to" kind of information, usually consisting of a record of possible motor and behavioral skills. Typical examples of Memory Objects in the Procedural Memory are behavioral rules.

In its current implementation, the *Procedural Memory* is a memory of rules in SOAR. They are stored in a text file and are loaded during MECA initialization process.

The *Working Memory* is internally split into many sub-memories: the Current Perception Sub-memory, the Cue Memory, the Episodic Recall Memory, the Imagination Sub-memory, the Global Workspace, the Predicted Situation Sub-memory, the Goals Sub-memory, the Plans Sub-memory, the Executive Plan Sub-memory and the Next Action Sub-memory.

The *Current Perception Sub-memory* holds a representation of objects identified in the environment by the *Perceptual Subsystem* in *System 1*, and is shared with the Episodic Subsystem. The *Cue Memory* is a memory holding cues to be used to query the *Episodic Memory*. The *Episodic Recall Memory* holds the episodes retrieved from *Episodic Memory*. The *Imagination Sub-memory* holds the structures required for the process of planning. The *Global Workspace* is a representation of information which was considered to be important enough (see sub-section 3.3.4, and interacts with the *Consciousness Subsystem*. The *Predicted Situation Sub-memory* is generated by the *Expectation Subsystem* and holds a representation of an immediate future. The *Goals Sub-memory* stores the goals generated by the *System 2 Motivational Subsystem*, and which are the starting point for the generation of plans. The *Plans sub-memory* holds all the plans generated by the *Planning Subsystem* and which were not yet executed. The *Executive Plan Sub-memory* holds a plan from the *Plans Sub-memory* which was decided to be the plan in execution in a given moment. And the *Next Action Sub-memory* is the next action to be taken while pursuing the plan in the *Executive Plan* sub-memory. This *Next Action Sub-memory* is then sent to the *Behavior and Motor Subsystem* in *System 1* in order to interfere with the automatic behavior being generated by the *System 1 Motivational Subsystem*.

3.3.3. The System 2 Motivational Subsystem

The *System 2 Motivational Subsystem* complements the *System 1 Motivational Subsystem* in providing MECA with Motivated Behavior. It is depicted in figure 11.

The sole purpose of *System 2 Motivational Subsystem* is providing the *Planning Subsystem* with the *Goals* required for the Planning process to start. *Goals* are the dual of

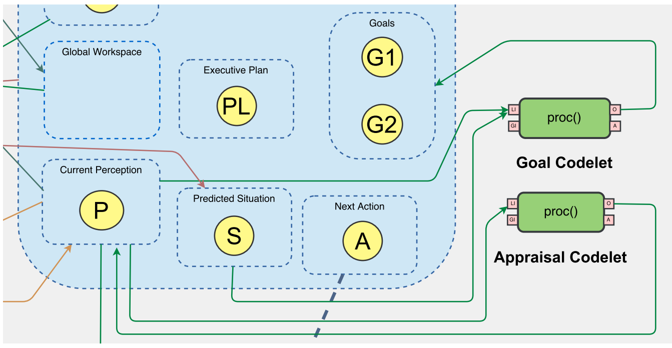

Figure 11: **A Zoom on the System 2 Motivational Subsystem**

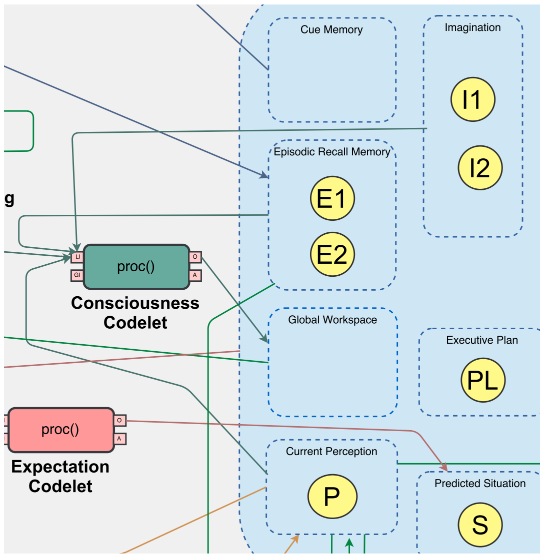

Figure 12: **A Zoom on the Consciousness Subsystem**

Drives in *System 1*, but are meant to a different purpose. Both *Drives* and *Goals* are meant to represent the *needs* (or purpose) for motivated behavior. But while *Drives* represent long-term needs, which are meant to be always under seek by the agent, *Goals* represent short-term needs, states which are to be timely reached and while they are reached, we say they were satisfied. So, a main difference between *Drives* and *Goals* is that *Drives*, despite being always being pursued, are never satisfied, while *Goals* should be satisfied at some point. This means we need to represent in *Goals* the conditions leading to this satisfaction.

Goals are created in MECA by the *Goal Codelet*. The *Goal Codelet* uses the *Current Perception* and the *Predicted Situation* in order to generate goals. Basically, it explores the space of possible future states, using an evolutionary technique, and selecting desirable future states as *Goals*.

A second component of the *System 2 Motivational Subsystem* is the *Appraisal Codelet*. This codelet is used to evaluate the *Current Perception* and tag it with with a value, which is then used by the Goal codelet to generate Goals.

3.3.4. The Consciousness Subsystem

The *Consciousness Subsystem* is depicted in figure 12.

Consciousness has emerged in animals apparently around 500 million years ago (Feinberg and Mallatt, 2013). One of the theories to explain the evolutionary advantage brought by consciousness is that consciousness works like a filter in the manifold of perceptions gathered by perception processes. This is basically the *Global Workspace Theory* (GWT) from Baars (1988). This filtered information is supposed to be the most relevant and important information at the present time for the animal in question. In *Global Workspace Theory*, this filtered information is then broadcast to all other subsystems, allowing an interesting dynamics to emerge. So, the *Global Workspace* is a privileged space within Working Memory, where very important information is supposed to be present.

The Global Workspace is a virtual kind of memory. Instead of storing its own set of Memory Objects, the Global Workspace is just a collection of references to other Memory Objects stored in the different memories described before, tagged as currently important.

Even though the topic of machine consciousness is still very controversial in the community (Gamez, 2009), one of the most popular approaches involves the implementation of *Global Workspace Theory*, which was implemented in the LIDA cognitive architecture (Franklin et al., 2012), and also by others (Shanahan, 2006; Dubois et al., 2008). In CST, the consciousness mechanism is not a built-in mechanism, but a mechanism which is implemented by means of consciousness codelets. It is true that these codelets make use of features provided by CST core, like the global input in codelets, which allow the broadcast required in GWT. The current implementation of CST provides a set of codelets which implements GWT in a way very similar to LIDA, but with some differences. In LIDA, the codelets assumed to be in a coalition are those which trigger at the same time. This is not the same in CST. In CST, codelets are assumed to be in a coalition just if they are coupled together by means of a common memory object. CST implementation of GWT also allows for subtle variations or interpretations of GTW, something which is not available in LIDA. An example on the use of consciousness codelets to implement GWT machine consciousness in a cognitive architecture using CST is given in Paraense et al. (2016b).

In the case of MECA, we are using a small variation of this mechanism, by means of the *Consciousness Subsystem*. Instead of promoting a competition among all unconscious sources of information, the consciousness mechanism in MECA selects information from only three different sub-memories: the *Current Perception*, the *Episodic Recall Memory* and the *Imagination*. At each time step, the *Consciousness Codelet* evaluates the Memory Objects within these three locations and choose something to send to the Global Workspace. The information in the Global

Workspace can then be used in the Planning Subsystem. The information in GWT is also used to feed the Episodic Memory, being collected by the Episodic Learning Codelet. This scheme allows for not only perception being stored in Episodic Memory, but also the contents of Imagination and Remembrance (the system can remember that in the past it remembered something).

3.3.5. The Expectation Subsystem

The expectation subsystem is illustrated in figure 13.

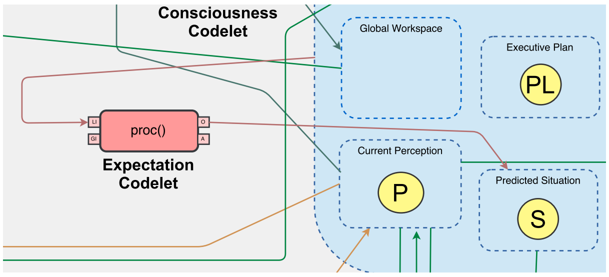

Figure 13: **A Zoom on the Expectation Subsystem**

In 2007, Jeff Hawkins published his book called "On Intelligence", (Hawkins and Blakeslee, 2007), where he discussed a new understanding on the phenomena of intelligence, based on the capacity of doing small predictions of future states. According to Hawkins, the brain is always performing these small predictions, and these are used to anticipate things which are imminent to happen. This ability might be crucial for doing things like catching a ball coming in our direction or deciding to dodge while a football player comes in our direction and we don't want to shock with him. According to Hawkins this ability is due to cortical microcircuits (George and Hawkins, 2009) which use to happen in the cerebral cortex. His theory gave rise to a whole field of research in neural networks, which he called *Hierarchical Temporal Memories* (HTM) (George, 2008; George and Hawkins, 2009).

The *Expectation Subsystem* in MECA is basically an attempt to include predictive abilities within our cognitive architecture. By this time, it is most a specification for something we consider to be important within a cognitive architecture, but still to be developed. An *Expectation Codelet*, supposed to run some sort of HTM like algorithm, will get information from the Working Memory (with all of its sub-memories), and should be generating a *Predicted Situation*, which is to be compared in the near future with the *Current Perception* in order to evolve its predictive abilities. Right now, the implementation of the Expectation Codelet is just a template for a future to be implemented algorithm based on HTM.

3.4. The OWRL Representation Language

Language is one of the unique capabilities of human beings, while compared to other cognitive abilities shared with other species of animals (Deacon, 1998). Recently, evidences that there are two subsystems in the brain responsible for language were discovered (Ardila, 2011), one responsible for grounding the meaning of isolated symbols (or words) and the other responsible for what is called *grammatical language*. One of our goals with MECA is to be able to use language to allow our MECA enabled agent to interact through language with other MECA enabled agents, or either human beings, such that they are able to communicate, with actual *understanding* in such communications.

In order to implement this understanding, we rely on *Grounded Cognition* to build *mental simulations* of a state of affairs happening at the environment. In its current stage of development, we defined a limited, but nevertheless quite elaborate ontology of known concepts which are able to be understood by a MECA mind. This constitutes the ontology of concepts which can populate the Working Memory, and compose what we call the OWRL - Object World Representation Language. An UML Diagram describing the ontology of concepts in OWRL is shown in figure 14, which was inspired on the *Conceptual Spaces* theory (see section 2.4).

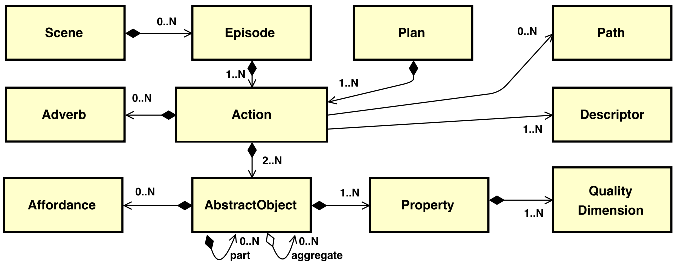

Figure 14: **An Ontology of Concepts in OWRL**

According to the OWRL ontology, the environment is represented as being a world full of *AbstractObjects*. *AbstractObjects* can be primitive, aggregate or composite. Primitive objects are defined by the list of its *Properties* and eventually *Affordances*. Composite objects can also have parts, which are also *AbstractObjects*. Aggregate objects are collections of *AbstractObjects* which are viewed as a single *AbstractObject*. A property can be define by one or more *Quality Dimensions*. For example, the property *Position* might have the quality dimensions X and Y. Affordances model possibilities of actions upon an *AbstractObject*. Affordances allow abstract objects to have a dynamics over time. This dynamics implies either in a change of some of its properties along time, new aggregate (or composite) objects to be created, or its aggregate or composite sub-objects to be destroyed. An instant of reality can be captured by a **configuration**, i.e. an *AbstractObject* collecting all perceived environment objects as its aggregate, in an instant of time. A configuration represents the reality state of affairs in this particular instant. We can represent the passage of time by a sequence of configurations, sampled in a periodic rate. A sequence of configurations where some important change happens is said to be an *Action*. An action can be represented alternatively by a *Descriptor* and a *Path*. A descriptor is a list of commands in OWRL which from a given configuration, can determine the final configuration obtained. Actions might have modifiers, which are modeled as *Adverbs*. A sequence of actions, might define an *Episode*. Usually, episodes are used to model a small number of objects which are part of the action. There might be the case in which many episodes are running at the same time. The union of all these episodes is a *Scene*. *Plans* are simply sequences of Actions.

Basically, this comprises the *Object World*. Right now, this is everything a MECA mind is able to understand from the environment. MECA's subsystems are able to start from sensors, acquiring specific quality dimensions, and perform higher abstractions, discovering World Objects, Episodes and Scenes. Instances of the many concepts pointed out at figure 14 are the entities populating the Working Memory, and the Planning Subsystem is able to create imaginative scenes with possible movements using all these elements. Goals are merely a specific configuration the system wants to achieve.

The next step in this research involves now the use of MECA to implement the ability of *Language*. The study on the simulation of language evolution has brought the attention on the importance of *Language Games* in order to construct the meaning of language in artificial agents (Steels, 2015; Vogt, 2015). In the next phase of this research project, we will be trying to use MECA to implement Language Games, allowing a MECA mind to represent its environment, learn its dynamics and use symbols to refer to the representation of both its understanding of reality, and possible alternative realities generated by imagination. An example of such developments, which helped us in setting up this framework is reported in (de Paula and Gudwin, 2015).

4. Conclusion

This work presented the specification and implementation details of a first version of the MECA Cognitive Architecture, with the aid of CST - The Cognitive Systems Toolkit and SOAR.

The system architecture was fully specified, and a first Java implementation was generated and is now under tests. We have built many small demo programs using MECA, but only using System 1 and System 2 as isolated subsystems, controlling traffic signals in a urban traffic simulation using the SUMO simulator (Krajzewicz et al., 2012). In order to evaluate more complex scenarios, involving the interaction between System 1 and System 2, we are currently working in a more extensive demo, which we hope to be the subject of a future work, with a full evaluation of the MECA architecture. The full implementation of the architecture will continue together with the next phase of our research project, where we will use the MECA package to implement a Cognitive Manager aimed at controlling generic IOT (Internet of Things) devices. In the next project stage, we will be addressing specifically the scenario of more complex urban traffic control, where the collaboration and cooperation between traffic lights, and their engagement in a language game (Steels and Hild, 2012) are of the utmost importance, aiming at taking advantage of the MECA cognitive architecture to provide a high level intelligent control of such environment.

Acknowledgments: The authors thank RLAM Innovation Center, Ericsson Telecomunicações S.A. Brazil (Proc. FUNCAMP 4881.2) and CEPID/BRAINN (Proc. FAPESP 2013/07559-3) for supporting this research.

5. References

Ardila, A., 2011. There are two different language systems in the brain. Journal of Behavioral and Brain Science 1 (2), 23–36.

Arkin, R. C., 1998. Behavior-based robotics. MIT press.

Augello, A., Infantino, I., Lieto, A., Pilato, G., Rizzo, R., Vella, F., 2016. Artwork creation by a cognitive architecture integrating computational creativity and dual process approaches. Biologically Inspired Cognitive Architectures 15, 74–86.

Baars, B. J., 1988. A Cognitive Theory of Consciousness. Cambridge University Press.

Baars, B. J., Franklin, S., 2007. An architectural model of conscious and unconscious brain functions: Global workspace theory and ida. Neural Networks 20 (9), 955–961.

Baddeley, A., 1997. Human Memory - Theory and Practice, revised edition Edition. Psychology Press, Taylor and Francis Group.

Baddeley, A., 2000. The episodic buffer: a new component of working memory? Trends in Cognitive Sciences 4 (11), 417–423.

Baddeley, A., 2002. Is working memory still working? European Psychologist 7 (2), 85–97.

Baddeley, A., Thomson, N., Buchanan, M., 1975. Word length and the structure of short-term memory. Journal Of Verbal Learning And Verbal Behavior 14 (6), 575–589.

Barsalou, L. W., 1999. Perceptual symbol systems. Behavioral and brain sciences 22 (04), 577–660.

Barsalou, L. W., 2008. Grounded cognition. Annu. Rev. Psychol. 59, 617–645.

Barsalou, L. W., 2010. Grounded cognition: Past, present, and future. Topics in cognitive science 2 (4), 716–724.

Bates, J., et al., 1994. The role of emotion in believable agents. Communications of the ACM 37 (7), 122–125.

Bergen, B., Chang, N., Narayan, S., 2004. Simulated action in an embodied construction grammar. In: Proceedings of the Twenty-Sixth Annual Conference of the Cognitive Science Society. pp. 108–13.

Braver, T. S., Cohen, J. D., Nystrom, L. E., Jonides, J., Smith, E. E., Noll, D. C., 1997. A parametric study of prefrontal cortex involvement in human working memory. NeuroImage (1), 49–62.

Brooks, R., 1986. A robust layered control system for a mobile robot. IEEE journal on robotics and automation 2 (1), 14–23.

Budakova, D., Dakovski, L., 2006. Computer Model of Emotional Agents. Vol. 4133. Springer Berlin / Heidelberg, p. 450.

Cabeza, R., Ciaramelli, E., Olson, I. R., Moscovitch, M., 2008. The parietal cortex and episodic memory: an attentional account. Nature Reviews Neuroscience 9 (8), 613–625.

Canamero, D., 1997. A hormonal model of emotions for behavior control. VUB AI-Lab Memo 2006.

Canamero, D., 1998. Issues in the design of emotional agents. In: Emotional and Intelligent: The Tangled Knot of Cognition. Papers from the 1998 AAAI Fall Symposium. pp. 49–54.

Cangelosi, A., Riga, T., 2006. An embodied model for sensorimotor grounding and grounding transfer: Experiments with epigenetic robots. Cognitive science 30 (4), 673–689.

Chella, A., Frixione, M., Gaglio, S., 2005. Planning by imagination in cicerobot, a robot for museum tours. In: Proceedings of the AISB 2005 symposium on next generation approaches to machine consciousness: imagination, development, intersubjectivity, and embodiment. pp. 40–49.

Cowan, N., 2001. The magical number 4 in short-term memory: a reconsideration of mental storage capacity. Behavioral and Brain Sciences 24, 87–114; discussion 114–85.

de Paula, S. M., Gudwin, R. R., 2015. Evolving conceptual spaces for symbol grounding in language games. Biologically Inspired Cognitive Architectures 14, 73–85.

Deacon, T. W., 1998. The symbolic species: The co-evolution of language and the brain. WW Norton & Company.

Dominey, P. F., Hoen, M., Inui, T., 2006. A neurolinguistic model of grammatical construction processing. Journal of Cognitive Neuroscience 18 (12), 2088–2107.

Dubois, D., Gaha, M., Nkambou, R., Poirier, P., 2008. Cognitive tutoring system with" consciousness". In: Intelligent Tutoring Systems. Springer, pp. 803–806.

Evans, J. S. B., 2003. In two minds: dual-process accounts of reasoning. Trends in cognitive sciences 7 (10), 454–459.

Evans, J. S. B., 2008. Dual-processing accounts of reasoning, judgment, and social cognition. Annu. Rev. Psychol. 59, 255–278.

Evans, J. S. B., Stanovich, K. E., 2013. Dual-process theories of higher cognition advancing the debate. Perspectives on psychological science 8 (3), 223–241.

Faghihi, U., Estey, C., McCall, R., Franklin, S., 2015. A cognitive model fleshes out kahneman's fast and slow systems. Biologically Inspired Cognitive Architectures 11, 38–52.

Feinberg, T. E., Mallatt, J., 2013. The evolutionary and genetic origins of consciousness in the cambrian period over 500 million years ago. Frontiers in Psychology 4.

Frank, M. J., Loughry, B., OReilly, R. C., 2001. Interactions between frontal cortex and basal ganglia in working memory: a computational model. Cognitive, Affective and Behavioral Neuroscience 1 (2), 137–60.

Frank, S. L., Vigliocco, G., 2011. Sentence comprehension as mental simulation: an information-theoretic perspective. Information 2 (4), 672–696.

Frankish, K., 2010. Dual-process and dual-system theories of reasoning. Philosophy Compass 5 (10), 914–926.

Franklin, S., Kelemen, A., McCauley, L., 1998. Ida: A cognitive agent architecture. In: Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on. Vol. 3. IEEE, pp. 2646–2651.

Franklin, S., Madl, T., D'mello, S., Snaider, J., 2014. Lida: A systems-level architecture for cognition, emotion, and learning. Autonomous Mental Development, IEEE Transactions on 6 (1), 19–41.

Franklin, S., Strain, S., Snaider, J., McCall, R., Faghihi, U., 2012. Global workspace theory, its lida model and the underlying neuroscience. Biologically Inspired Cognitive Architectures 1, 32–43.

Gamez, D., 2009. The potential for consciousness of artificial systems. International Journal of Machine Consciousness 1 (02), 213–223.

Gärdenfors, P., 2004. Conceptual spaces: The geometry of thought. MIT press.

Gärdenfors, P., 2014. The geometry of meaning: Semantics based on conceptual spaces. MIT Press.

George, D., 2008. How the brain might work: A hierarchical and temporal model for learning and recognition. Ph.D. thesis, Stanford University.

George, D., Hawkins, J., 2009. Towards a mathematical theory of cortical micro-circuits. PLoS Comput Biol 5 (10), e1000532.

Goertzel, B., Pennachin, C., Geisweiller, N., 2014a. Brief survey of cognitive architectures. In: Engineering General Intelligence, Part 1. Atlantis Press, pp. 101–142.

Goertzel, B., Pennachin, C., Geisweiller, N., 2014b. Engineering General Intelligence, Part 2: The CogPrime Architecture for Integrative, Embodied AGI. Atlantis Thinking Machines. Atlantis Press.
URL https://books.google.com.br/books?id=a9-RngEACAAJ

Gudwin, R. R., 2015. Computational semiotics: The background infrastructure to new kinds of intelligent systems. APA Newsletter - Philosophy and Computers 15 (1), 27–38.
URL http://www.apaonline.org/resource/collection/EADE8D52-8D02-4136-9A2A-729368501E43/ComputersV15n1.pdf

Hamadi, Y., Jabbour, S., Saïs, L., 2010. Learning for dynamic subsumption. International Journal on Artificial Intelligence Tools 19 (04), 511–529.

Hawkins, J., Blakeslee, S., 2007. On intelligence. Macmillan.

Heckel, F. W., Youngblood, G. M., 2010. Multi-agent coordination using dynamic behavior-based subsumption. In: Sixth Artificial Intelligence and Interactive Digital Entertainment Conference - AIIDE.

Hofstadter, D., Mitchell, M., 1994. The Copycat Project: A Model of Mental Fluidity and Analogy-making. BasicBooks, 10, East 53rd Street, New York, NY 10022-5299, pp. 205–268.

Howard, M. W., Fotedar, M. S., Datey, A. V., Hasselmo, M. E., 2005. The temporal context model in spatial navigation and relational learning: toward a common explanation of medial temporal lobe function across domains. Psychological Review 112 (1), 75–116.

Hull, C., 1943. Principles of behavior.

Jilk, D. J., Lebiere, C., O'Reilly, R. C., Anderson, J. R., 2008. Sal: An explicitly pluralistic cognitive architecture. Journal of Experimental and Theoretical Artificial Intelligence 20 (3), 197–218.

Kahneman, D., 2011. Thinking, fast and slow. Farrar, Strauss and Giroux, New York / Macmillan.

Kotseruba, I., Gonzalez, O. J. A., Tsotsos, J. K., 2016. A review of 40 years of cognitive architecture research: Focus on perception, attention, learning and applications. arXiv preprint arXiv:1610.08602.

Krajzewicz, D., Erdmann, J., Behrisch, M., Bieker, L., December 2012. Recent development and applications of SUMO - Simulation of Urban MObility. International Journal On Advances in Systems and Measurements 5 (3&4), 128–138.

Laird, J., 2012. The Soar cognitive architecture. MIT Press.

Lallee, S., Madden, C., Hoen, M., Dominey, P. F., 2010. Linking language with embodied and teleological representations of action for humanoid cognition. Frontiers in neurorobotics 4.

Langley, P., Laird, J. E., Rogers, S., 2009. Cognitive architectures: Research issues and challenges. Cognitive Systems Research 10 (2), 141–160.

Lieto, A., Chella, A., Frixione, M., 2016. Conceptual spaces for cognitive architectures: A lingua franca for different levels of representation. Biologically Inspired Cognitive Architectures.

Lieto, A., Radicioni, D., Rho, V., 2017. Dual peccs: A cognitive system for conceptual representation and categorization. Journal of Experimental & Theoretical Artificial Intelligence 29(2), 433—452.
URL http://dx.doi.org/10.1080/0952813X.2016.1198934

Lucentini, D. F., Gudwin, R. R., 2015. A comparison among cognitive architectures: A theoretical analysis. Procedia Computer Science 71, 56–61.

Madden, C., Hoen, M., Dominey, P. F., 2010. A cognitive neuroscience perspective on embodied language for human–robot cooperation. Brain and language 112 (3), 180–188.

Marques, H. G., Holland, O., 2009. Architectures for functional imagination. Neurocomputing 72 (4), 743–759.

Maslow, A. H., 1943. A theory of human motivation. Psychological review 50 (4), 370.

McCall, R. J., 2014. Fundamental motivation and perception for a systems-level cognitive architecture. Ph.D. thesis, The University of Memphis.

McNab, F., Klingberg, T., 2008. Prefrontal cortex and basal ganglia control access to working memory. Nature Neuroscience 11 (1), 103–107.

Meyer, J. J. C., 2006. Reasoning about emotional agents. Int. J. Intell. Syst 21 (6), 601–619.

Miller, G., 1956. The magical number 7, plus or minus two: Some limits on our capacity for processing information. Psychological Review 63.

Nakashima, H., Noda, I., 1998. Dynamic subsumption architecture for programming intelligent agents. In: Proceedings of the 3rd

International Conference on Multi Agent Systems - ICMAS. IEEE Computer Society, pp. 190–197.
Narayanan, S., 1999. Moving right along: A computational model of metaphoric reasoning about events. In: AAAI/IAAI. pp. 121–127.
Osman, M., 2004. An evaluation of dual-process theories of reasoning. Psychonomic bulletin & review 11 (6), 988–1010.
Paraense, A. L. O., Raizer, K., de Paula, S. M., Rohmer, E., Gudwin, R. R., 2016a. The cognitive systems toolkit and the cst reference cognitive architecture. Biologically Inspired Cognitive Architectures 17, 32–48.
Paraense, A. L. O., Raizer, K., Gudwin, R. R., 2016b. A machine consciousness approach to urban traffic control. Biologically Inspired Cognitive Architectures 15, 61 – 73.
URL http://www.sciencedirect.com/science/article/pii/S2212683X15000614
Pezzulo, G., Calvi, G., 2011. Computational explorations of perceptual symbol systems theory. New Ideas in Psychology 29 (3), 275 – 297, special Issue: Cognitive Robotics and Reevaluation of Piaget Concept of Egocentrism.
URL http://www.sciencedirect.com/science/article/pii/S0732118X09000336
Picard, R. W., 1997. Affective Computing. MIT Press, Cambridge.
Reilly, W. S. N., 1996. Believable social and emotional agents. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA.
Rosenblueth, A., Wiener, N., Bigelow, J., 1943. Behavior, purpose and teleology. Philosophy of science 10 (1), 18–24.
Roy, D., 2005. Semiotic schemas: A framework for grounding language in action and perception. Artificial Intelligence 167 (1), 170–205.
Samsonovich, A. V., 2010. Toward a unified catalog of implemented cognitive architectures. In: Proceedings of the First Annual Meeting of the BICA Society. BICA Society, IOS Press, Amsterdam, The Netherlands, pp. 195–244.
Septseault, C., Nédélec, A., 2005. A model of an embodied emotional agent. In: IVA. p. 498.
Setchi, R., Lagos, N., Froud, D., 2007. Computational imagination: research agenda. In: AI 2007: Advances in Artificial Intelligence. Springer, pp. 387–393.
Shanahan, M., 2006. A cognitive architecture that combines internal simulation with a global workspace. Consciousness and cognition 15 (2), 433–449.
Sloman, S. A., 1996. The empirical case for two systems of reasoning. Psychological bulletin 119 (1), 3.
Stanovich, K. E., West, R. F., 2000. Individual differences in reasoning: Implications for the rationality debate? Behavioral and Brain Sciences (23), 645–726.
Steels, L., 2015. The talking heads experiment. Computational Models of Language Evolution. Language Science Press.
Steels, L., Hild, M., 2012. Language grounding in robots. Springer Science & Business Media.
Sun, R., 2003. A tutorial on clarion 5.0. http://www.cogsci.rpi.edu/public_html/rsun/sun.tutorial.pdf.
URL http://www.cogsci.rpi.edu/public_html/rsun/sun.tutorial.pdf
Sun, R., 2009. Motivational representations within a computational cognitive architecture. Cognitive Computation 1 (1), 91–103.
Sun, R., Wilson, N., 2011. Motivational processes within the perception–action cycle. In: Perception-Action Cycle. Springer, pp. 449–472.
Tulving, E., 1991. Concepts of human memory. In: Squire, L., Lynch, G., Weinberger, N., McGaugh, J. (Eds.), Memory: Organization and locus of change. Oxford Univ. press, pp. 3–32.
Tulving, E., 2002. Episodic memory: From mind to brain. Annual Review of Psychology 53, 1–25.
Vogt, P., 2015. How mobile robots can self-organise a vocabulary. Computational Models of Language Evolution.