THE AUTONOMOUS SYSTEMS LABORATORY
UNIVERSIDAD POLITÉCNICA DE MADRID

Systems, Models and Self-Awareness

Towards Architectural Models of Consciousness

Ricardo Sanz, Carlos Hernández
Jaime Gómez, Julia Bermejo, Manuel Rodríguez
Adolfo Hernando, Guadalupe Sánchez

ASLab A-2009-016 v 1.0 Final
February 10, 2010

Abstract: The lack of a well established theory of consciousness is a major difficulty in the construction of machines that express some of the functionalities associated to it. The ASys long term project intends the construction of assets for engineering any-scale, self-aware autonomous systems. A core element in this effort is the proposal of an architecture for consciousness of general applicability. In this article we analyze some of the current problems present in the construction of machines based on cognitive models and propose a model-based strategy to enhance the rigor of the theorization on consciousness and its mapping into realizations in technical systems.
Keywords: Consciousness, machine consciousness, systems engineering, cognitive models, model-driven engineering.

1 Introduction

The construction of conscious machines is hampered by two major problems: 1) the lack of a solid theory of biological consciousness to be used as baseline in the engineering of the machines and 2) the inherent difficulties of engineering complex and less than well understood systems (mainly because of 1). These two problems are the *What* and *How* problems of machine consciousness.

It is in this problematic context where we propose a change of strategy compared with current practices in cognitive robotics. Instead of using the exploratory programming methods that are common in the field, we propose to fall back into model-driven engineering methods in search of a way for properly do the engineering and, at the same time, explicitly capturing the basic essences of the architecture that will indeed constitute a theory of consciousness of applicability in domains beyond robotics. If done with proper generality and ambition, this may even constitute a solid contribution to the advance of a general theory of consciousness of applicability to biological cognition. To what extent this may constitute a valid explanation of consciousness for philosophers Bengtsson (2003) we don't really care.

The structure of the article is as follows: fist we will present a rationale for a systems engineering strategy in machine consciousness (MC) research; then we will analyze a major problem in the mapping of cognitive models into realizations; after that we will describe the model-based systems engineering strategy and outline its application in the engineering of conscious machines.

2 Engineering conscious machines

There are many possible different strategies to try to build conscious machines. Some of them will be self-organizing approaches, but due to the complexity of the problem and the dimensions of the associated configuration/design space these strategies are far from being of any possible impact today.

The other major alternative are design-based approaches. Among all these, two major strategies (as described by Holland some time ago) are what may be called the *direct theory-driven* and the *incremental exploratory* strategies to MC:

Direct approaches: These approaches try to build a machine following a particular theory of consciousness (e.g. Baars GWT Baars (1997)) leading to the construction of concrete designs typically restricted to a concrete class of implementation technologies Shanahan (2006) Franklin (2000).

Incremental approaches: This second strategy tries to add competencies one-at-a-time up to reaching the level of consciousness. This strategy is mostly driven by the conviction that consciousness is a compound phenomenon that emerges from the interaction of more basic competences. In these cases, the specific technological niche of application is what determines the competences needed and hence the concrete stepping to MC Holland (2003) Sloman (2009).

We may get the impression that no matter what is the approach used, MC will suffer a receding-horizon phenomenon similar to what has been happening with AI. The argued failure of AI has been a failure to provide what some expected to

be a complete human intellect; but to our understanding this was not the program of AI, because this last was a more precise one of developing the required competences in solving problems that required human intelligence. The so-called strong AI vision lacks a precise, technical specification and hence there is not an ultimate test procedure for validation.

To avoid this receding horizon phenomenon in MC it is a best strategy to precisely state the conditions for the termination of the work, i.e. what engineers call the *requirements* for the system to be. In oder to achieve this, the best strategy is to follow a direct approach, i.e. specify the thing as a whole. This specification will be very abstract and centered in requirements at the beginning and will gain consistency and detail as it progresses through the engineering process. The specification shall be done in terms understandable to all stakeholders, that in the case of MC will be both engineers and cognitive scientists. This originates a specification language problems, as the domain languages of engineers are much more precise that what cognitive scientists in general are prepared to read.

Some may argue that a design approach to consciousness is doomed to failure. The main argument for this being the evolutionary nature of consciousness and the design-less nature of evolutionary processes. This argument does not hold, however, as it is based in the *evolutionary/developmental fallacy*: i.e. the belief that evolved/epigenetic systems can have different properties than designed ones.

The construction trajectory —the past— of a system does not affect what it can do; what a system can do is only determined by what the system is at the present state. So, given a certain class of systems, both evolutionary and design-based strategies can be used. The problem —the real problem— will be the actual factibility of the evolutionary/developmental/design-centric process that in the case of complex systems —as we expect MC to be— is always hampered by the dimension of the design space Gelsey et al. (1998).

2.1 On the need of systems engineering approach

The advances towards having the necessary competences for engineering self-aware, conscious machines can happen in two main ways: i) as an artisanal practice based on exploratory approaches to mimicking conscious performances of humans and animals or ii) as an engineering practice based on solid scientific theories.

If we decide to focus on this second approach, the theories needed must be quantitative and this confronts the mainstream theorization in the field of consciousness that is mainly descriptive when not purely metaphysical (in the worst sense of the term). Beyond the needs for engineering, this quantitative theoretical approach may help reaching a common scientific picture for the consciousness domain.

But, while there are some valuable attempts to mathematically formalize some theories of consciousness, they are always facing the problem of the generalized

difficulties in understanding mathematical formalism by consciousness research stakeholders.

The work in the ASys Project —and associated C3 and ICEA projects— addresses these issues by trying to be both precise and descriptive by means of using formal and semiformal engineering modeling languages. One of the core aspects to be modeled that affects the whole engineering life-cycle is the architecture of the system. For this class of modeling, the modeling language selected is SysML, a merger of the software-centric Unified Modeling Language and common modeling practices in the systems engineering community.

The rationale for this approach is simple: minds are complex systems and to engineer them we need complex systems engineering methods. Systems engineering is a general term to refer to all kind of activities related with the construction of systems but it is also the name of a concrete engineering strategy to enable the construction of maximal complexity systems INCOSE (2004). Systems engineering is a merger of conventional engineering activities and organization/operations research that enables the concurrent development of the many activities in a large project without losing the cohesion necessary to produce a solid system.

One of the main values of this strategy is the adoption of a multilevel, multiparadigm approach to system modeling that enables the collaboration of very heterogeneous stakeholders. This is obviously of maximal relevance to the field of MC.

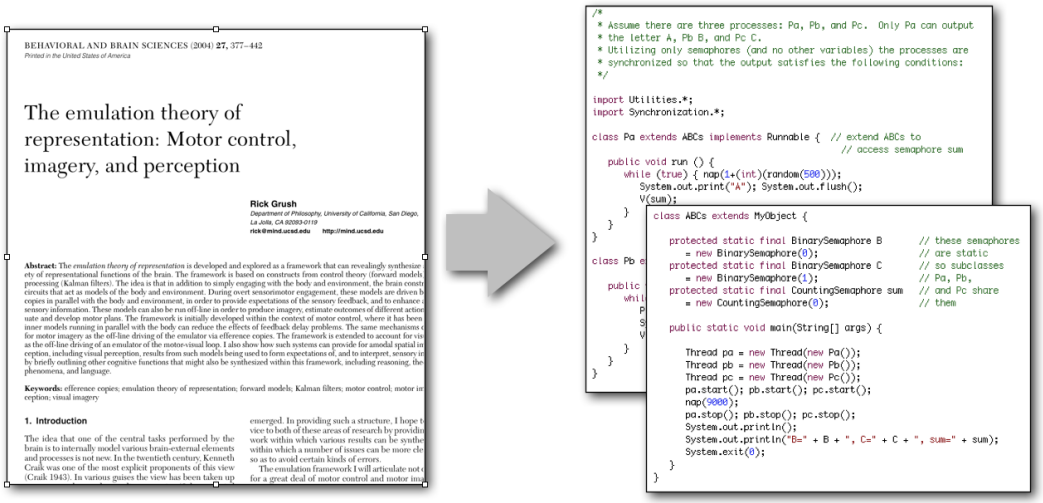

Figure 1: The basic strategy followed in the attempts to implement conscious systems follow the general strategy in the implementation of bio-inspired cognitive systems: a model is described in text and implemented —by human transformation— into code.

2.2 Building conscious machines: the very idea

Conscious machines are a particular subclass of the more general cognitive systems domain and their construction strategies are not different from them.

In most cases, cognitive systems are inspired in concrete cognitive competences of animals. In fact, for many researchers, cognitive science is just the study of the human mind and hence *cognitive system* is equated to the implementation of a concrete human mental competence. This implementation is usually referred to as a *model* of this competence McClelland (2009)Morse et al. (2008).

The basic approach to cognitive system construction can be summarized in the following steps:

1. The specification of a theory of the particular competence to be realized in the system.

2. The mapping of the theory to a computer based implementation.

3. The evaluation of the system in a concrete testbed to determine if the theory, when realized, renders the same class of performance as humans do.

This strategy is shown in Figure 2.1. The central objective of this work is usually the validation of concrete theories of human cognition. However, when a model matches a concrete data set associated with certain system, is not necessarily matching the inner structure of the modeled system. Passing the test only makes from the model a potential candidate for the explanation of the natural phenomenon. From the perspective of artificial systems Simon (1996) it is only the performance of the function what is relevant Newell (1990) (not the matching of the inner structure).

The use of the term *model* in this context is sometimes confusing. Cognitive scientists will use it for the implementation (the thing at the right of figure 2.1) because it is a model of the biological function. Artificiality engineers will use it for the theory —the design— as captured in a persistent form (the thing at the left of figure 2.1).

Modeling theory gives us precise definitions of model, and we will use the term for any concrete instance of any class that is in correlation with some entity of our interest —whether biological or technical— and can be used to provide answers about the modeled system.

The strategy depicted in figure 2.1 is also the common strategy followed in the domain of machine consciousness. The theories sustaining the realizations are captured in non-rigorous models and mapped into hardware/software-based implementations[1].

[1]The decisions concerning the mapping to hardware and/or software are the central topic of in-

Textual descriptions are usually described in natural language accompanied by what Shaw denominated boxology Shaw and Garlan (1996). From a model theoretic perspective, both text —with or without free form diagrams— and source code (left and right in figure 2.1) are models of the system to be and of the biological originator (if it exists). In general we will restrict the use of the term *model* for those representational stages prior to software/hardware implementation (see below).

2.3 Building conscious machines: the very reality

This being the commonly understood picture of cognitive systems engineering, the fact is that the reality of the construction of cognitive systems is somewhat different: the system that we build, run and evaluate is usually not the system described in the models.

There are several sources of mismatch and two of major importance: 1) The mapping from model to source code and 2) the mapping from source code to implementation.

The mapping from source code to implementation is a task done by compilers, linkers and operating systems. A lot of work is being done in this context in the domain of safety critical embedded systems to guarantee that the mapping is done with rigor, i.e. that source and target are actually functionally equivalents.

The mapping from model to code, however is our major concern here. The main problem is that, contrary to the source code, the abstract model guiding the construction of the cognitive system is an incomplete and non rigorous model. The transformation is done by hand, by a human —usually a graduate or postgraduate cognitive science student— using non automated procedures.

Obviously the programmer can introduce errors in the mapping from model to code, but what is worse is that in order to make the system work he must 1) fill-in the blanks and 2) invent some hacks to make the final system work.

The filling of blanks indicates that the model lacks some elements that are obviously necessary but not explicitly indicated in it (e.g. the types of codification used for the data). The injection of hacks is worse: it indicates that the theory is wrong. Here we are not describing a rare situation. This situation is well known to any implementer of cognitive systems. The mapped theory does not work as expected —i.e. the robot does not behave as expected— but some clever hacks in the machine can obviate the problem.

In some cases the hacks are back-propagated to the theory hence evolving it to more complete versions McClelland (2009) but in many other cases they are not and

terest for the embodied cognition movement Anderson (2003)Wilson (2002) but apart from the mechanisms that bridge the informational/physical frontier —sensors and actuators— there is not much relevance in what goes to software and what goes to hardware.

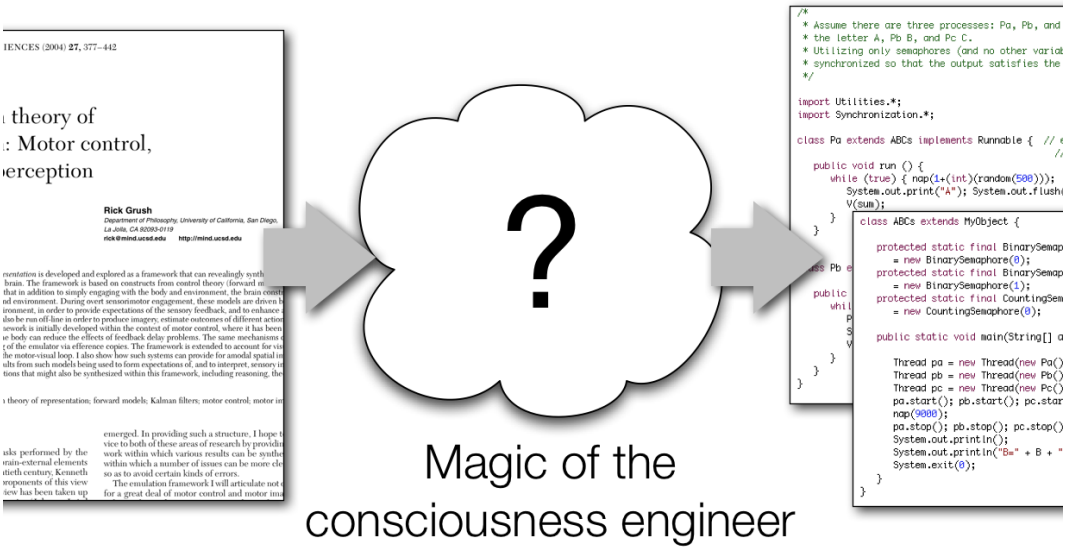

Figure 2: As the models captured in the textual descriptions are not complete, the basic strategy shown in 2.1 has to be complemented with some magic from the hands of the programmer. To make the cognitive system work some hacks shall be introduced in the code to make it minimally operative. Those hacks are usually not conceptualized and fed-back into the model that stays as it was. In the case of consciousness research the risk is even higher due to the elusive nature of the phenomenon.

the passing of the behavioral tests by the target robot is taken as an incontrovertible proof of the validity of the theory as it was.

In the case of MC this risk is even higher due to the difficult nature of the topic that escapes clear conceptualizations and thwarts the specification of rigorous testing procedures.

2.4 Towards a Positive Theory of Consciousness

The strategy proposed here intends precise modeling even in early stages. The main reason for this is to help resolve the issues derived from the too many basic understandings of what consciousness is: process, function, module, property, emergent phenomenon, quantum state, etc.

This problem can be traced back to a common architectural reverse-engineering problem: the extraction of function and structure from external observations of complex systems. As was said before, the observation or modeling of I/O behavior does not in general enable the extraction of system structure but just of system

function (in a mathematical sense Ljung (1998)).

This structure extraction problem has to be conjugated with the architectural insights that introspection can give, rendering a problem of data compliance with non-formal description of architectures.

A rapid scan of the literature on the topic leaves the impression that most theories of mind that target the whole thing seem just literature (or plain bullshit, or love & hate manifestations). Positive theories of mind and consciousness are mostly seem as partial and naïve.

In the search for universal MC we need a unified theory of consciousness that fulfills the following desiderata:

- The theory targets the *whole thing*, from access awareness and self awareness to phenomenological aspects of mind (i.e. even qualia).
- The theory is *widely agreed* across disciplines and to be so it must be understandable and explanatory[2].
- The theory is expressible in different abstraction levels that are in strict correspondence (cf. issues in philosophy of realization across levels) to be at the same time *general*, *precise* and *verifiable*.

We must express the unified theory of consciousness in a formal enough language as to minimize the problem of multiple interpretations derived from the multiple backgrounds, competences and objectives of theory stakeholders. Some approaches to formal theories of consciousness are already available in the literature but are far from being accepted because they are not targeting the whole (e.g. Tononi Tononi (2004) or Ehresmann Ehresmann and VanBremeersc (2002)), they are just ungraspable (e.g. Kirilyuk's Kirilyuk (2003) or Zeleznikar's Zeleznikar (1997)) or simply unjustified (e.g. quantum mechanic accounts). There's even some researchers that think that formalization of consciousness is untenable. But achieving a level of rigor is a necessary step in any consolidated science. It is our impression that the key to the formalization in mind theory is going to pass through very general system theorization frameworks (e.g. Klir's Klir (1969) or Mesarovic's Mesarovic and Takahara (1989)) or modeling and simulation frameworks (e.g. Ziegler et al. approaches Stevenson (2003)).

2.5 Modes of model expression

There are plenty of modes of expressing theories. In the domain of cognitive science the most common is to use the textual narratives that are the major vehicle

[2]This last being a true hard problem due to the different nature of explanations in the different disciplines. However, the model-based theory of explanation Craik (1943) can help solve this problem.

in psychology and philosophy. Verbal-linguistic models are interchanged sharing words and histories; documents that, while fully readable and even enjoyable, suffer a major problem of vagueness and hermeneutical lack of robustness. Textual documents seem to capture theories but the reality is far from consolidating them in a unified theoretical body. Why are we stuck here? The conclusion that in this domain the pressure for publishing novel theories overcomes the forces for unification seems credible and even more than plausible.

On the other side there are the logical-mathematical models used in physics and engineering disciplines. This may be realized in different ways —equations, drawings, MatLab models, etc— that share a common property: formal rigor. For some this rigor may be rigor mortis, when sacrificing the flexibility and generality of less constrained languages just for the sake of rigor.

Graphic-visual models using pictures, charts or drawings of any type are used by many people to add visual compaction to an otherwise complicated description. In many cases images enable the reduction of complexity that may arise in textual or mathematical descriptions, exploiting the enormous cognitive bandwidth of our visual system.

As an example of the conceptual complexity we are involved in, Figure 2.5 shows part of the taxonomy of cognitive emotions as proposed by Ortony Ortony et al. (1988).

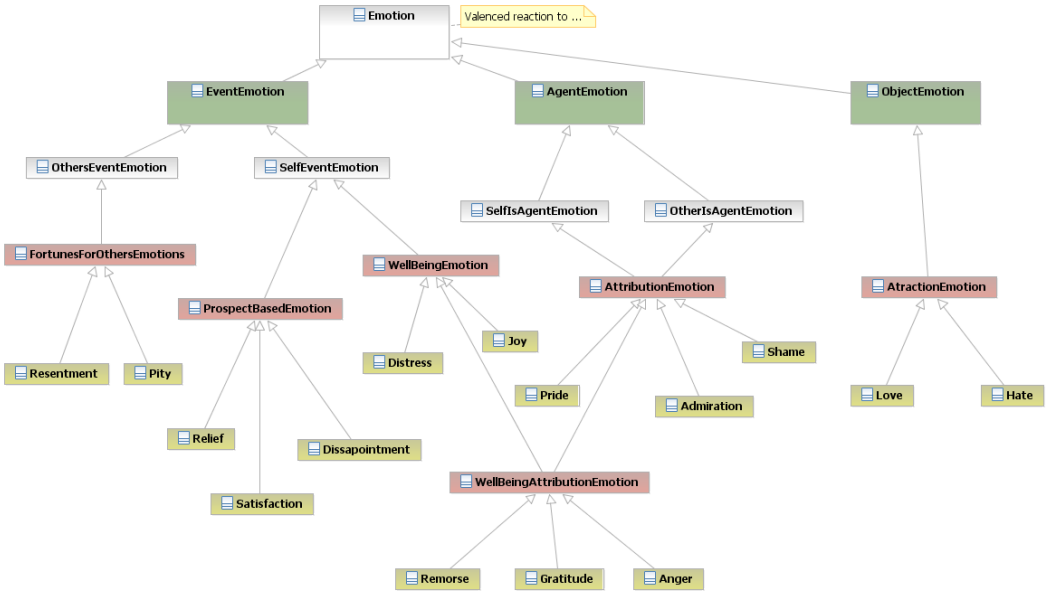

Figure 3: A partial taxonomy of cognitive emotions based on Ortony et al. (1988). The use of a graphic diagram let reduce the burden of explaining all these relations in a cumbersome narrative.

2.6 Rationale for a Systems Approach to Consciousness

The recognition of the enormous complexity of the issue of building conscious machines points into a direction that can help solve both problems. Complex systems engineering is addressed using the methods of the so-called discipline of *Systems Engineering*. This methodological tool sits in the middle of technical and managerial issues, addressing the multifaceted problems of large-scale, complex systems engineering.

In the ASys Project we propose the use of recent systems engineering modeling methods to address the complexities of MC theorizing and system synthesis. The core objective of this approach is the development of *Reference Models of Consciousness* in a systematic, shareable, rigorous way.

There are several aims in this proposal:

- Consolidate a unified vision on consciousness functions and mechanisms.
- Organize knowledge about consciousness components into standardized, reusable and extensible models.
- Develop methods for (re-)using this knowledge in support of the construction of conscious artificial systems.

This approach is obviously model-centric, with models playing many roles in it. Let's quote Rothemberg in an attempt to precisate the nature of models Rothenberg (1989):

> *Modeling, in the broadest sense, is the cost-effective use of something in place of something else for some cognitive purpose.*
> ...
> *A model represents reality for the given purpose; the model is an abstraction of reality in the sense that it cannot represent all aspects of reality. This allows us to deal with the world in a simplified manner, avoiding the complexity, danger and irreversibility of reality.*

The ASys proposed method to follow in this *Reference Models of Consciousness* approach consists in the following steps:

1. Define an ontology for describing consciousness components and systems.
2. Define reference models as standardised elements that include knowledge and information about the form, function and behavior of components.
3. Formalize the ontology and reference models in UML/SysML.

4. Create semantic mappings between UML/SysML, cognitive science, systems biology and engineering tools.

This approach can target the two problems described in the introduction —what and how— at once; because if the models built are rigorous enough, they can not only address problem 1 (*what* consciousness is) but can also help with problem 2 (*how* to build the thing).

This can be possible by the unification of the emerging model-based theory of consciousness and the modern model-based practice of embedded systems construction. Let's see some of the bricks of this approach.

3 On Model-based minds

The ASys Framework is a theoretical framework for cognition analysis developed to support the engineering of self-x systems Sanz et al. (2005). This framework specifies a model of cognitive processes based on the use of models in the generation of behavior.

To summarize it, cognitive systems interact with other systems in their environment by means of model based representations sustained by executable models statistically linked to the entities involved. This implies that the cognitive mind is an stochastic model-based controller, where explicit models are used in the many activities that a system involving a cognitive agent —subject+object— can be involved in: control, anticipation, postdiction, etc. This vision is in strong correlation with some theoretical biology positions Rosen (1985).

Examples of similar understandings abound in the literature, as for example in Shanahan Shanahan (2006):

> "Cotterill (1998, 2001) advances the proposal that thought is "internally simulated interaction with the environment," and Hesslow (2002) argues that this "simulation hypothesis" can explain our experience of an inner world."

Consider for example the question of meaning in perception Pustejovsky (1990) (see Figure 3). López López (2007) presents a rigorous model on the construction of internal models of objects in the environment based on formal mapping between the causally connected border quantities and the internal representative quantities. This theoretical model of perception is based on general systems theory Klir (1969) in an attempt to provide a theory of applicability both to the natural and the artificial.

In this framework, classic topics of cognitive science and philosophy are model-reframed. Knowledge is equated to executable dynamic models —models about

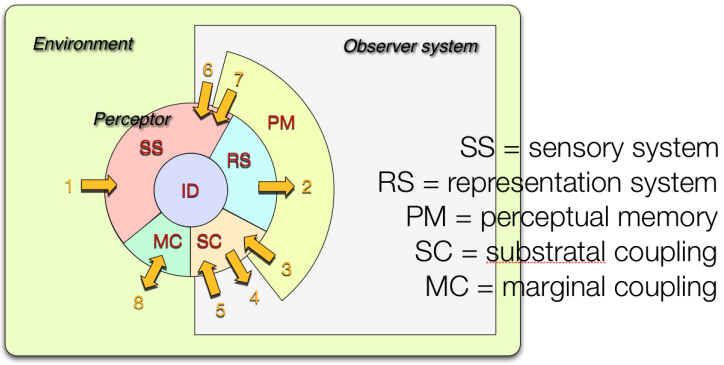

Figure 4: A general model of a perceptual system as suggested by López López (2007). The perceptual system maps the externally received information from objects in the environment into internal models that are aggregated into the perceptive memory (a perceptual model repository).

some (partial) reality in/out of the agent that may be executed— much in line with information-centric conceptions of knowledge Dretske (1981). The internalized models that the cognitive agent uses are executed over a physical execution engine (e.g. cerebellum) or over a virtual machine. Virtual machines are indeed models running over a physical execution engine or another virtual machine.

It is also possible that models do not appear in explicit form and may be degenerated (e.g. a simple static value) or they may be embodied (i.e. "precompiled" with the execution engine). Explicitness enable sharing among heterogeneous purpose execution engines.

This vision can be summarized in a single sentence: Minds are model-based controllers. However, their varieties and uses are enormous: they can be direct/inverse, implicit/explicit, static/dynamic, isolated/coupled, "genetic"/"memetic", homogeneous/heterogeneous, postdictive/predictive, etc.

4 On Model-based systems engineering

4.1 What is Systems Engineering?

Systems engineering is an interdisciplinary approach whose core objective is to enable the realization of successful complex systems INCOSE (2004). It focuses on precisely defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design, synthesis and system validation while considering the complete problem.

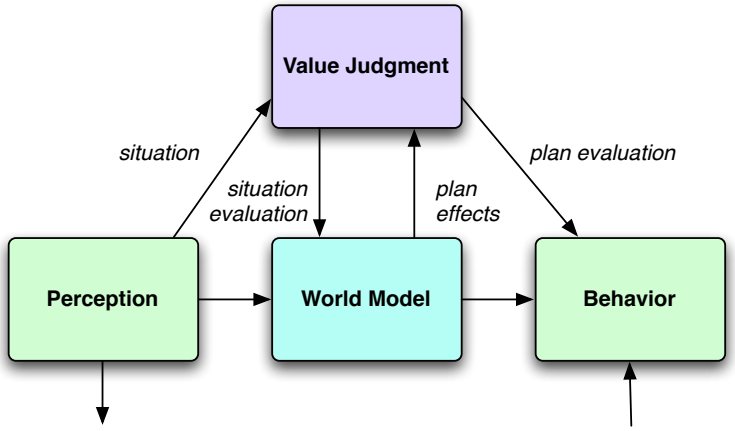

Figure 5: The ASys general model of a cognitive system reflects a common trend in theoretical analyses of cognition (e.g. the model proposed by Albus and Meystel Albus (1999)). The fundamental cognitive system is composed by a behavior generation engine driven by a model updated by a perceptual system and teleologically governed by a value system.

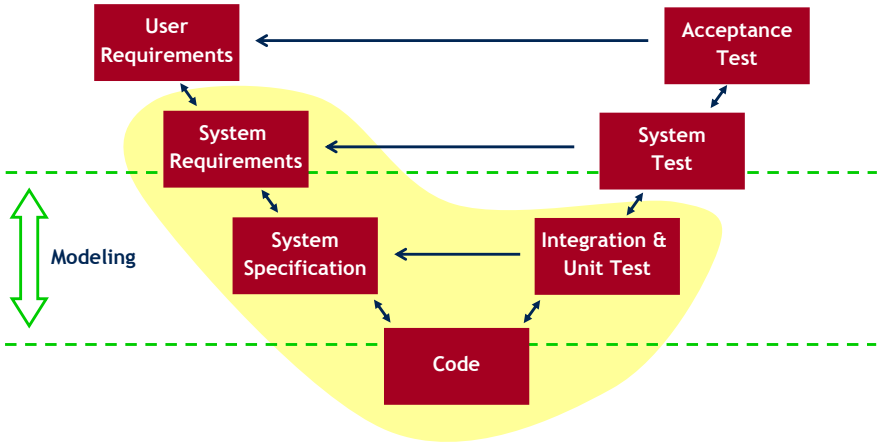

Figure 6: The V-model of development expresses a strategy of progressive completion of tasks and simultaneous task-level validation to avoid the excessive costs derived from detecting errors at the very end of the engineering process.

4.2 Does this match our needs?

Talking about ”customers” and ”needs” may seem far-off for research on machine consciousness. However, research is just another kind of *intentional* human activity and hence it pursues some goals of interest to certain communities.

In this field we can identify three classes of global research objectives for MC:

- Understanding *biological consciousness* is a major objective in cognitive science. Building of machines can help in this activity by clarifying the issues and building of computational models that are, in a sense, explanatory. These computational models, however shall be used with care as metaphors or theories of biological systems McClelland (2009).

- Building *machines like us* (the C3PO drive) is another thread of activity in machine consciousness strongly linked to the anthropomorphic thread in robotics. The ultimate reason for this is not clear however, from understanding ourselves (in the line of the previous drive), to building more usable machines (in the line of the next drive) sometimes it seems that human-mimetic robots are built just as a *playing god* game.

- Building *better machines* (the SkyNet drive) is the basic grounding of research in the purely technical side. The rationale here is basically that self-awareness will improve systems resilience Sanz et al. (2007a). In the context of human-machine interaction, there are also issues related to the usability of the system Picard (1997).

All them specify some needs for our realizations. The value of systems engineering is that it provides a systematic way of addressing those very heterogenous needs (aka requirements). The V-model of complex systems development (see Figure 4.1) helps into addressing this heterogeneity.

4.3 A Focus on Software and Architecture

The focus of the ASys project is to streamline the construction of autonomous systems by focusing on their software-intensive aspects and the use of their architecture as the core asset guiding all the process. This enables the exploitation of domain oriented assets in a variety of application domains Sanz et al. (1999).

The specification of the architecture of a system can't be done in the technology of the final implementation due to the natural mismatch in abstraction levels. For this purpose different kinds of modeling languages are used that go from the most general and versatile to the most precise and rigorous.

In this context, *natural language* is a normal vehicle of communication between scientists and in fact the almost only vehicle of communication in the most philosophical aspects of cognitive science. Hence most descriptions of issues related to higher level aspects of cognition —consciousness in particular— are done in plain text (see Figure 4.3). This is done so because natural language can be accommodated to almost any kind of need; but this flexibility is also its main problem because this leads to continuous misunderstandings in the use of the terms that cannot be resolved by reverting to other more restricted forms of meaning conveyance.

Here metaphors play a critical role Davidson (2001) but they are also a source of divergences in conceptualizations.

XYZ addresses complex systems by means of scalable design patterns. This approach is specially well captured in the multiresolutional approach fostered by the control design pattern that Meystel calls the elementary loop of functioning (Meystel, 2003). Of importance in relation with the ASys theory of meaning is the incorporation of value judgment mechanisms over this elementary loop.
The elementary loop of functioning, when applied hierarchically, generates a multiresolutional ladder of meanings specifically focused on the controllable subspace of each control level. This approach partitions both the problem of meaning generation and the problem of action determination, leading to hierarchical control structures that have interesting properties of self-similarity.
This core design pattern approach is extended in the concept of a control node of the RCS control architecture (Albus, 1992). Beyond the model of the world and the sensing and acting units, this architecture considers the existence of a value judgment unit that evaluate both static states and dynamic states derived from hypothetical plan execution.

Figure 7: Natural language is a normal vehicle of communication between scientists and in fact the almost only vehicle of communication in the most philosophical aspects of cognitive science.

4.4 On soft and hard block models

Beyond text, the normal form of capturing a mechanistic model Bechtel (2007) in cognitive science is to use any kind of graphical diagram. These diagrams are representations of reverse-engineered hypothetical structures as a graph of labeled nodes (”boxes”) and connections between them (as lines or arrows).

The idea of boxes and arrows representation is very useful because many problems in modular systems design are reducible to boxes and arrow between them. However the excessive freedom of this language can be counterproductive. The term boxology Shaw and Garlan (1996) refers pejoratively to this free-form nature of these diagrams that usually mislead readers to beliefs of deeper understanding. See for example the diagram shown in Figure 4.4; the heterogeneous labeling of the boxes and variety of arrows is a clear indication of lack of rigor.

In the domains of software engineering multiple more precise languages have evolved in a pursuit to capture the essence of a design; from the most elementary levels of programming data types definition and composition to the higher levels of system architecture description Krutchen (1995).

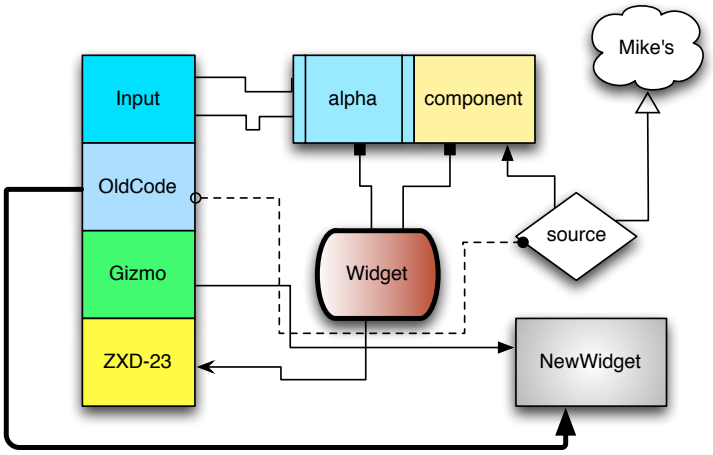

Figure 8: A "boxology" diagram tries to capture some structural understanding of a system. However, bounded syntactic rules are usually necessary to make the diagram an effective vehicle of communication.

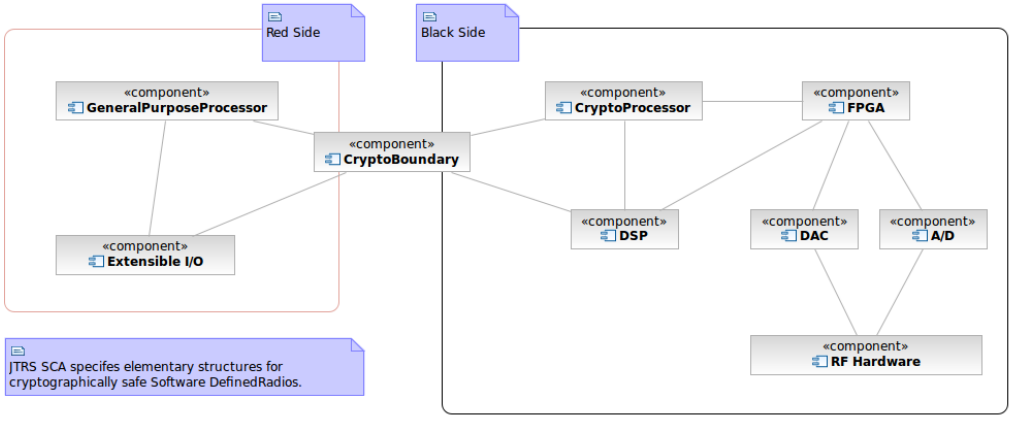

Figure 9: As an example or an increased precision diagram, an UML component diagram captures structural properties of a modular system following bounded syntactic rules that makes the diagram a precise vehicle of communication between stakeholders.

4.5 Beyond models as human languages

Obviously, all the modeling languages described so far are languages written and used by humans. When the rigor is augmented and the language is formalized it can be read and written also by machines. This is the case of programming languages.

The model-driven development movement in engineering Balmelli et al. (2006) strives for the use of formal models to capture engineering designs. The models shall be complete and rigorous enough as to be usable in the automated generation of systems. This is may be of major difficulty in the case of physical systems Gershenfeld (2005) but it is well known known in the world of software systems Aho et al. (2006).

However, the use of programming languages requires a strict education of humans in the use of a very non-natural language that induces also problems of understandability and shareability. It is difficult to understand a program written by another programmer.

In this context, the model-driven development strategy fosters more accessible while rigorous modeling that can bridge the gap human-machine, and human-human and still be usable as technical assets as programming languages are. An example of this strategy is the well known OMG Model Driven Architecture approach OMG (2003).

The final conclusion of all this discussion on models, is that in cognitive science, and in particular when very abstract issues are at stake, precise communication is key.

The engineer and the scientist must use a consistent, well-defined, and well-understood language to communicate the requirements and design to other stakeholders (engineers, scientists or not), otherwise the product will be questionable, founder, fail, or be a full disaster. For the software-intensive systems engineer, today that language is UML/SysML.

5 The ASys vision of self-aware machines

Model-based control is a well established domain inside the automatic control technology spectrum. For example, model predictive control (MPC) Camacho and Bordons (2007) has been in effective use in the process industries —e.g. refineries or chemical plants— for the last twenty+ years. MPC uses dynamic models of the process under control to optimally compute the best action in relation to a concrete future horizon.

Internal model controllers exploit models of the systems —the bodies— they are controlling and also of the part of the world that constitutes their environment. This enables the solution of the inverse problem of control action determination (going from final states to initiating actions). In the case of biological systems these models may come ontogenetically, learned or culturally transferred.

Due to the complexity of the realizations of sophisticated controllers, there are new sources of disturbances that affect not only the body but the mind of the cognitive agent. In these conditions there is an increased need of implementing mecha-

nisms for applying feedback and feedforward competences to the controllers themselves. Figure 5 shows the basic structure of a model-based controller extended with two new competences: 1) the model now includes a model of itself[3] and 2) the continuous modeling competences are extended to address the issues of this self-model.

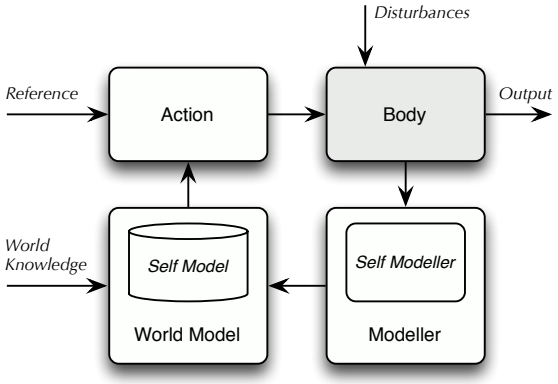

Figure 10: The "self" extension adds a self model and the continuous modeling competences that are necessary to handle the updating of this self-model.

The key of the ASys approach to self-modeling systems is using for this role of self-models, the rigorous models used in the engineering of the system itself. The central idea is hence to break the design-time / run-time barrier concerning the modeling of the cognitive system itself. This *'Self' step* will enable the systems to have cognitive access to its very implementation.

The core concepts of the model-based self-awareness vision have been captured elsewhere in the form of a set of principles for conscious systems engineeringSanz et al. (2007b):

Model-based cognition: A cognitive system exploits models of other systems in their interaction with them.

Model isomorphism: An embodied, situated, cognitive system is as good performer as its models are.

Anticipatory behavior: Except in degenerate cases, maximal timely performance is achieved using predictive models.

Unified cognitive action generation: Generate action based on an integrated, scalable, unified model of task, environment and self in search for global performance maximization.

[3] In this context, the body of the agent can be considered part of the environment.

Model-driven perception: Perception is realized as the continuous update of the integrated models used by the agent in a model-based cognitive control architecture by means of real-time sensorial information.

System awareness: An aware system is continuously perceiving and generating meaning -future value- from the continuously updated models.

System self-awareness: A conscious system is continuously generating meanings from continuously updated self-models in a model-based cognitive control architecture.

System consciousness: The cognitive system experiments qualia.

6 The Systems Modeling Language

In order to implement this vision it is necessary to find modeling languages that are rigorous enough as to be used to build models of the cognitive system that can be used as technical assets in a model-driven engineering process France and Rumpe (2007). At the same time and due to the still premature stage of understanding about consciousness wee need a language with some ontological flexibility.

These two reasons lead us to selecting the UML/SysML languages to implement this vision. The Unified Modeling Language (UML) is a well known language used in software and process engineering OMG (2009)Scholz-Reiter et al. (2007). The the Systems Modeling Language (SysML) is a modeling language for holistic system representation and systems engineering OMG (2008)Balmelli (2007).

SysML is a recent specification of a graphical/textual, semi-formal modeling language addressing the issues of the Systems Engineering RFP (Request fro Proposals) developed by the OMG, INCOSE, and AP233. It can be seen as a UML Profile that represents a subset of UML2 with some non-UML additional extensions to support the specification, analysis, design, verification and validation of systems that include hardware, software, data, personnel, procedures, and facilities.

Some aspects of SysML are the following:

- SysML is broader than software-centric modeling languages.
- It can capture salient aspects of complex system design.
- It is quite intuitive for system engineers, supporting proven systems engineering concepts like requirements, hierarchical block structuring and parametrics.
- The language has been designed to not be a barrier to traditional system engineering methods.

The SysML constructs are diagrams and textual notation that can be used to address the early modeling of a wide range of systems. The notation is simple and powerful, being oriented to complex engineering problems. It has been specially designed to be particularly effective in specifying requirements, structure, behavior, and allocations and constraints on system properties to support engineering analysis. SysML reuses a subset of the UML 2 assets, mainly diagrams. A summary of the SysML diagrammatic taxonomy can be seen in Figure 6.

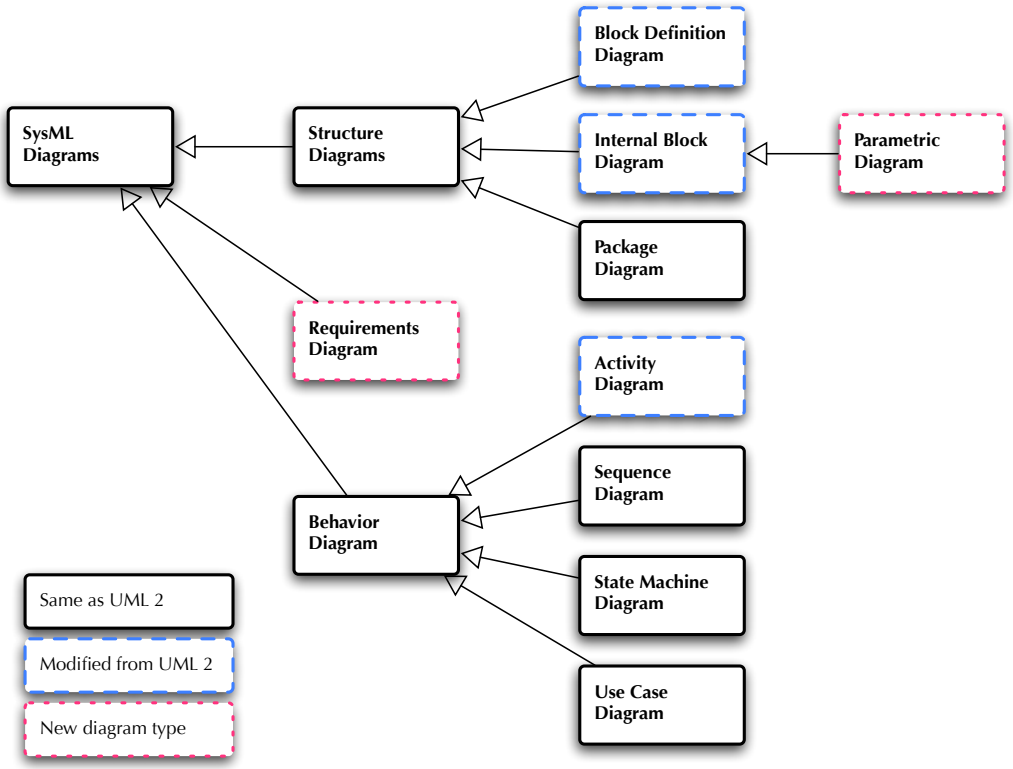

Figure 11: The SysML diagram set includes some of the UML diagrams (some untouched and others modified for systems engineering) and some new diagrams specially important for requirements-driven systems engineering activities (from OMG (2008)).

7 Towards an UML/ SysML model of consciousness

The ASys approach based on model-based systems engineering can be described as explicit model-based, reflective, predictive, adaptive autonomous systems engineering. The major value is that autonomous control based on self-models may enable an increased awareness that can eventually lead to more robust autonomous

performance, providing a road to both 1) expressing an unified theory of consciousness and 2) using it to build machines.

As was said before, the key strategy is to use the engineering models as self-models. The lack of complete formality in UML/SysML models will imply some necessary model-to-model and model-to-text transformations of the UML/SysML models into other data representations to be used by the model exploitation engines. This for example has been demonstrated in the domain of autonomic computing Trencansky et al. (2006)Calinescu (2007).

This is an ongoing work being developed in the ICEA and C3 projects and focusing on three concrete aspects:

- A deep control engineering, theoretical model of self-awareness.
- A model-based construction process based on this engineering model.
- An architecture for model-based autonomous systems exploiting these models.

7.1 The tooling: Rational Systems Developer

To follow a model-driven engineering process for complex systems it is necessary to employ the adequate tools that can implement and coordinate the many activities that are required. In this work we have selected the IBM Rational Systems Architect (RSA) tool suite to handle the complexity of systematic cognitive system engineering.

IBM RSA leverages the power of Eclipse, enabling the extension of the development environment by the inclusion of new plugins that may be necessary to exploit some classes of cognitive models (e.g. neural-network models). This also enables the simultaneous development of code from UML 2 to C/C++, Java and CORBA-based applications.

We expect to leverage RSA forward and reverse transformation capabilities to help automate the transition between models and code enabling an exploratory development project.

7.2 The ongoing modeling effort

The current modeling effort is concentrated of the elaboration of the core ontology for autonomous systems (OASys) and the modeling of the central architectural construct of this theory: the Epistemic Control Loop (See Figure 7.2).

The epistemic control loop focuses on the model-centric organization of cognitive control systems, identifying four classes of activities around the central models

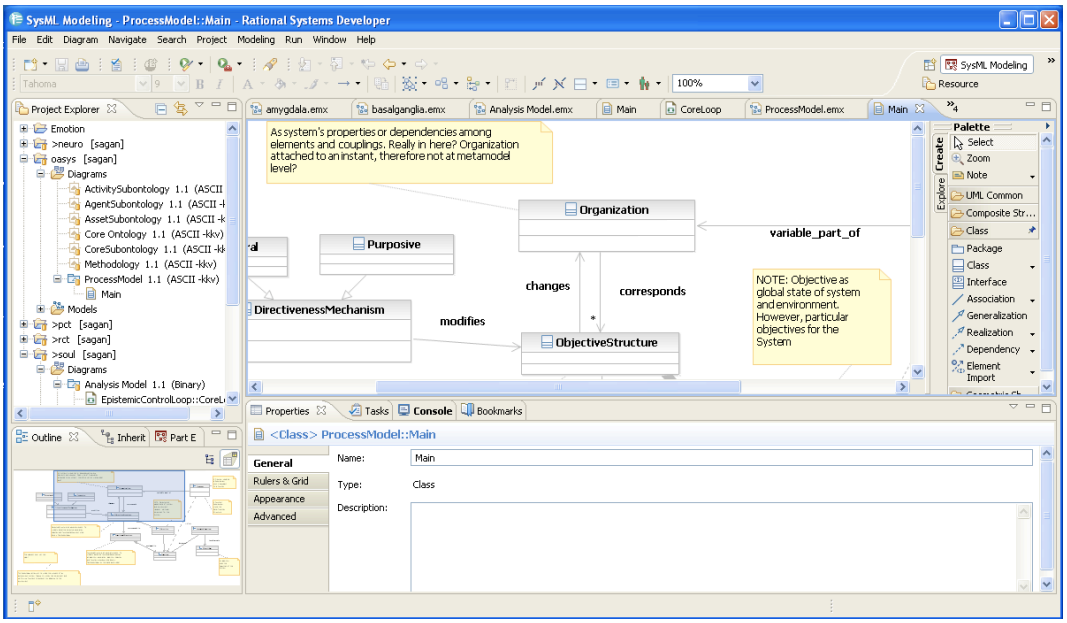

Figure 12: Rational Software Architect (RSA) is the tooling selected for supporting exploratory model-driven development of the many assets necessary for the realization of the ASys Vision.

(action, perception, value and model transformation). This model is in strong correlation with Albus and Meystel elementary loop of functioning Albus (1995) or Gudwin knowledge units Gudwin (2002).

7.3 Model Structure

The ASys model is structured in several submodels:

The ASys Ontology model: a collection of core concepts for the general autonomous systems domain. This ontology is strongly based on General Systems Theory to be of applicability both in th engineering of systems and in the explanation of biological phenomena.

The SOUL Architecture model: a model of a general architecture for an autonomous self-aware autonomous agent.

The Domain models: models of domain-specific character (in the current work centered on brain, robotics and process control)

The application models: concrete models of final technical systems under development. The RCT is a mobile robotics testbed and the PCT is a continuous process control testbed.

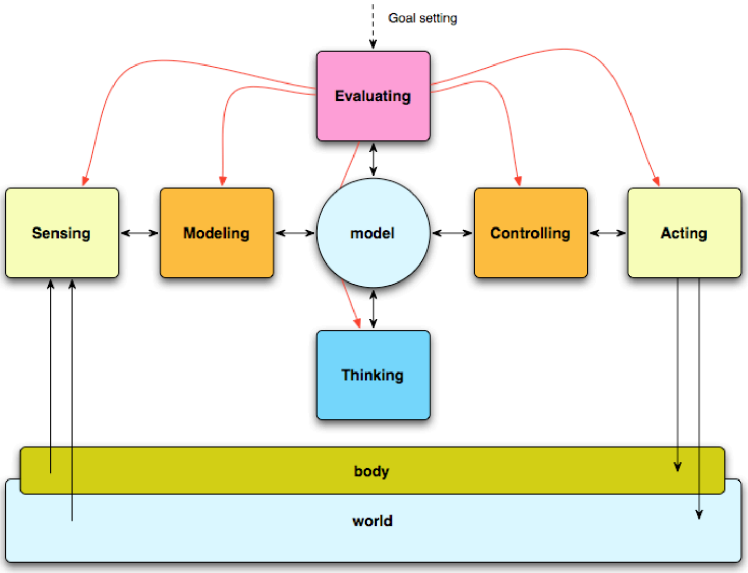

Figure 13: The epistemic control loop focuses on the model-centric organization of cognitive control systems. Four classes of activities are identified around the central models (action, perception, value and model transformation).

8 Summary

The core objective of the ASys approach, and in particular of the work described here, is the development of architectural reference models for cognitive systems. Of central importance is the development of a Reference Model for Consciousness (the SOUL model) that can help both explain natural phenomena of consciousness and direct the development of technical systems Godfrey-Smith (2006).

The approach is very ambitious as it intends an ultimate explanation of consciousness. Consilience of the variegated perspectives of consciousness may seem unfeasible in principle due to their very different nature Dale et al. (2009) but we consider that the main problem underlying this variety is the lack of a powerful enough set of concepts that can cover all the phenomenon Aydede and Güzeldere (2005). If we are successful in this approach some consciousness theories will be necessarily abandoned but some of them shall be unified into a more solid construct.

The model-based science Magnani et al. (2002); Magnani (2006) approach taken here is based on two pillars: general systems theory and model-driven engineering. The construction of model-driven engineering models —using engineering languages like UML/SysML— will enable the expression of the model-based control theory of consciousness in a formal enough language as to minimize the problem of multiple interpretations. These languages are simple enough as to be un-

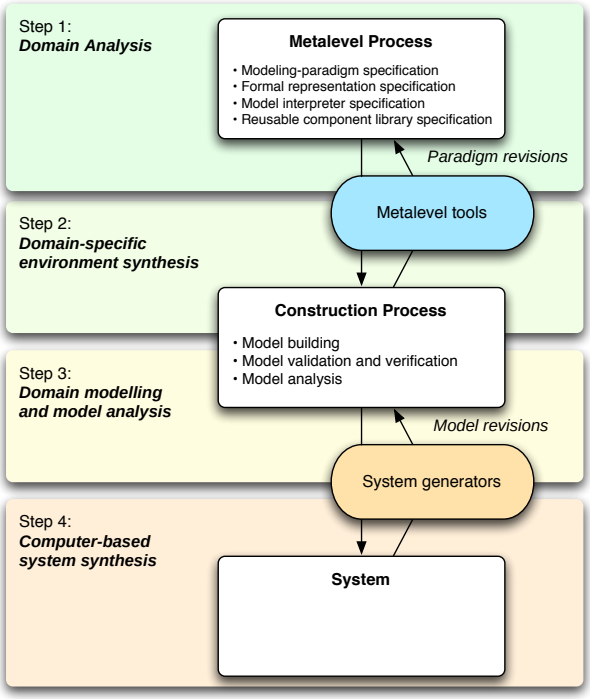

Figure 14: The domain engineering approach taken in the ASys project follows a four step process. Current work involves mainly metalevel processes, the provision of metalevel tools and systems generators, and initial model building.

derstandable by a variety of stakeholders. The existence of the models in this form will also straightforward the development of systems based on this architecture.

The point of theoretical convergence is the role that models play in cognition and specially in consciousness aspects and how these very models can be fully equated with traditional and non-traditional conceptions of knowledge Ford et al. (1993). This re-gains the strategy for model-centric cognitive-science that was started by Craik Craik (1943).

To conclude this article let's include a quote from Sommerhoff Sommerhoff (1990):

> "the various obstacles that confront those who seek to deal with consciousness in a physical language can be overcome if a strictly methodical approach is followed in which from the start all *analytical concepts are accurately defined* in physical terms."

9 Acknowledgements

We acknowledge the support of the Spanish Ministry of Education and Science through grant *C3: Control Consciente Cognitivo* and the European Commission thorough Grant *ICEA: Integrating Cognition, Emotion and Autonomy*.

This article is based on a talk given at the NOKIA Workshop on Machine Consciousness 2008, that took place in Helsinki, Finland from August 21 to August 22. We want to acknowledge the role that Pentti Haikonen has played in the preparation of these concrete materials and Antonio Chella in the launch of the IJMC forum for machine consciousness.

References

Aho, A. V., Lam, M. S., Sethi, R., and Ullman, J. D. (2006). *Compilers: Principles, Techniques, and Tools*. Addison Wesley, 2nd edition edition.

Albus, J. S. (1995). RCS: A Reference Model Architecture for Intelligent Systems. In *AAAI 1995 Spring Symposium on Lessons Learned from Implemented Software Architectures for Physical Agents*.

Albus, J. S. (1999). The Engineering of Mind. *Information Sciences*, 117(1-2):1–18. http://www.isd.mel.nist.gov/documents/albus/engineeringmind-96.pdf.

Anderson, M. L. (2003). Embodied cognition: A field guide. *Artificial Intelligence*, 149:91–130.

Aydede, M. and Güzeldere, G. (2005). Cognitive architecture, concepts, and introspection: An information-theoretic solution to the problem of phenomenal consciousness. *Noûs*, 39(2):197–255.

Baars, B. J. (1997). In the theatre of consciousness. global workspace theory, a rigorous scientific theory of consciousness. *Journal of Consciousness Studies*, 4:292–309.

Balmelli, L. (2007). An overview of the systems modeling language for products and systems development. *Journal of Object Technology*, 6(6):149–177.

Balmelli, L., Brown, D., Cantor, M., and Mott, M. (2006). Model-driven systems development. *IBM Systems journal*, 45(3):569–585.

Bechtel, W. (2007). *Mental Mechanisms: Philosophical Perspectives on Cognitive Neuroscience*. Lawrence Erlbaum.

Bengtsson, D. (2003). The nature of explanation in a theory of consciousness. Technical Report LUCS 106, Lund University Cognitive Studies.

Calinescu, R. (2007). Model-driven autonomic architecture. In *ICAC'07. Fourth International Conference on Autonomic Computing*, Jacksonville, Florida, USA.

Camacho, E. F. and Bordons, C. (2007). *Model Predictive Control*. Springer, second edition.

Craik, K. J. (1943). *The Nature of Explanation*. Cambridge University Press.

Dale, R., Dietrich, E., and Chemero, A. (2009). Explanatory pluralism in cognitive science. *Cognitive Science*, 33(5):739–742.

Davidson, D. (2001). What metaphors mean. In *Inquiries into Truth and Interpretation*, pages 245–264. Oxford University Press, Oxford, UK.

Dretske, F. I. (1981). *Knowledge and the flow of information*. MIT Press, Cambridge, Mass.

Ehresmann, A. and VanBremeersc, J. (2002). Consciousness as structural and temporal integration of the context. Online.

Ford, K. M., Bradshaw, J. M., Adams-Webber, J. R., and Agnew, N. M. (1993). Knowledge acquisition as a constructive modeling activity. In Widman, L. E., Loparo, K. A., and Nielsen, N. R., editors, *Knowledge acquisition as modeling*, pages 9–32. John Wiley & Sons, Inc., New York, NY, USA.

France, R. and Rumpe, B. (2007). Model-driven development of complex software: A research roadmap. In *Future of Software Engineering, 2007. FOSE '07*, pages 37–54.

Franklin, S. P. (2000). Building Life-Like 'Conscious' Software Agents. *Artificial Intelligence Communications*, 13:183–193.

Gelsey, A., Scwabacher, M., and Smith, D. (1998). Using modeling knowledge to guide design space search. *Artificial Intelligence*, 101(1-2):35–62.

Gershenfeld, N. (2005). *FAB: The Coming Revolution on Your Desktop–From Personal Computers to Personal Fabrication*. Basic Books.

Godfrey-Smith, P. (2006). The strategy of model-based science. *Biol Philos*, 21:725–740.

Gudwin, R. R. (2002). Semiotic synthesis and semionic networks. *S.E.E.D. Journal (Semiotics, Evolution, Energy, and Development)*, 2(2):55–83.

Holland, O., editor (2003). *Machine Consciousness*. Imprint Academic, Exeter, UK.

INCOSE (2004). Systems engineering handbook. Technical Report INCOSE-TP-2003-016-02, International Council on Systems Engineering.

Kirilyuk, A. P. (2003). Emerging consciousness as a result of complex-dynamical interaction development. In *Workshop on Machine Consciousness: Complexity Aspects*, Turin, Italy.

Klir, G. C. (1969). *An Approach to General Systems Theory*. Van Nostrand Reinhold.

Krutchen, P. (1995). The "4+1" view of software architecture. *IEEE Software*, 12(6):42–50.

Ljung, L. (1998). *System Identification: Theory for the User*. Prentice Hall PTR, 2nd edition.

López, I. (2007). *A Framework for Perception in Autonomous Systems*. PhD thesis, Departamento de Automática, Universidad Politécnica de Madrid.

Magnani, L., editor (2006). *Model Based Reasoning in Science and Engineering*. College Publications.

Magnani, L., Nersessian, N. J., and Pizzi, C., editors (2002). *Logical and Computational Aspects of Model-Based Reasoning*. Kluwer Academic Publishers, Dordretch, The Neterlands.

McClelland, J. L. (2009). The place of modeling in cognitive science. *Topics in Cognitive Science*, 1(1):11 – 38.

Mesarovic, M. and Takahara, Y. (1989). *Abstract Systems Theory*. Springer-Verlag, Berlin.

Morse, A. F., Lowe, R., and Ziemke, T. (2008). On he role(s) of modelling in cognitive science. *Pragmatics and Cognition*, 16(1):37–56.

Newell, A. (1990). *Unified Theories of Cognition*. Harvard University Press.

OMG (2003). *MDA Guide*. Object Management Group, version 1.0.1 edition.

OMG (2008). *OMG SysML Specification*. Object Management Group, version 1.1 edition.

OMG (2009). *Unified Modeling Language (UML)*. Object Management Group, version 2.2 edition.

Ortony, A., Clore, G. L., and Collins, A. (1988). *The Cognitive Structure of Emotions*. Cambridge University Press.

Picard, R. W. (1997). *Affective Computing*. The MIT Press.

Pustejovsky, J. (1990). Perceptual semantics: the construction of meaning in artificial devices. In *5th IEEE International Symposium on Intelligent Control*, volume 1, pages 86–91.

Rosen, R. (1985). *Anticipatory Systems*. Pergamon Press.

Rothenberg, J. (1989). The nature of modeling. In Widman, L. E., Loparo, K. A., and Nielsen, N. R., editors, *Artificial intelligence, simulation & modeling*, pages 75–92. John Wiley & Sons, Inc., New York, NY, USA.

Sanz, R., Alarcón, I., Segarra, M. J., de Antonio, A., and Clavijo, J. A. (1999). Progressive domain focalization in intelligent control systems. *Control Engineering Practice*, 7(5):665–671.

Sanz, R., López, I., and Bermejo-Alonso, J. (2007a). A rationale and vision for machine consciousness in complex controllers. In Chella, A. and Manzotti, R., editors, *Artificial Consciousness*. Imprint Academic.

Sanz, R., López, I., Bermejo-Alonso, J., Chinchilla, R., and Conde, R. (2005). Self-x: The control within. In *Proceedings of IFAC World Congress 2005*.

Sanz, R., López, I., Rodríguez, M., and Hernández, C. (2007b). Principles for consciousness in integrated cognitive control. *Neural Networks*, 20(9):938–946.

Scholz-Reiter, B., Kolditz, J., and Hildebrandt, T. (2007). UML as a basis to model autonomous production systems. In Cunha, P. F. and Maropoulos, P. G., editors, *Digital Enterprise Technology. Perspectives and Future Challenges*, pages 553–560. Springer, New York, USA.

Shanahan, M. (2006). A cognitive architecture that combines internal simulation with a global workspace. *Consciousness and Cognition*, 15(2):433–449.

Shaw, M. and Garlan, D. (1996). *Software Architecture. An Emerging Discipline*. Prentice-Hall, Upper Saddle River, NJ.

Simon, H. A. (1996). *The Sciences of the Artificial*. MIT Press, Cambridge,USA, third edition.

Sloman, A. (2009). An alternative to working on machine consciousness. *International Journal of Machine Consciousness*, V(N).

Sommerhoff, G. (1990). *Life, Brain And Consciousness: New Perceptions through Targetted Systems Analysis*. Elsevier Science Publishing Company, 1 edition.

Stevenson, D. (2003). From DEVS tpo formal mehtods: A categorical approach. In *Proceedings of SCSC Conference*.

Tononi, G. (2004). An information integration theory of consciousness. *BMC Neuroscience*, 5(42).

Trencansky, I., Cervenka, R., and Greenwood, D. (2006). Applying a UML-based agent modeling language to the autonomic computing domain. In *OOPSLA '06: Companion to the 21st ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*, pages 521–529, New York, NY, USA. ACM.

Wilson, M. (2002). Six views of embodied cognition. *Psychonomic Bulletin & Review*, 9(4):625–636.

Zeleznikar, A. P. (1997). Informational theory of consciousness. *Informatica (Slovenia)*, 21(3).

Contents

Title: Systems, Models and Self-Awareness
Subtitle: Towards Architectural Models of Consciousness
Author: Ricardo Sanz, Carlos Hernández
Jaime Gómez, Julia Bermejo, Manuel Rodríguez
Adolfo Hernando, Guadalupe Sánchez
Date: February 10, 2010
Reference: A-2009-016 v 1.0 Final
URL: http://www.aslab.org/public/documents/A-2009-016.pdf

Autonomous Systems Laboratory

UNIVERSIDAD POLITÉCNICA DE MADRID
C/JOSÉ GUTIÉRREZ ABASCAL, 2
MADRID 28006 (SPAIN)