A Rationale for Artificial Consciousness

Ricardo Sanz

Autonomous Systems Laboratory
Universidad Politécnica de Madrid
The very first

Thanks to the organisers for inviting me to this workshop

Apologise for using this opportunity to present bold, half baked ideas and ongoing work behind our research
Contents

- Introduction to complex control
- Approaches to autonomy
- Modelling approach to development
- Model-based self-aware control systems
Introduction
This speech is a reflection, that describes some core ideas behind long term research in complex control systems.

The objective of this research is to provide architecture-centric, component based technology for the construction of high autonomy control systems.
from dreams ...
Forbidden Planet

Robbie

(1956)
2001: A Space Odyssey (1968)
Star Wars

C3PO
R2D2

(1977)
Blade Runner

Roy
Pris
Zhora
Leon

(1982)
Terminator

T800

(1984)
I Robot

Sonny

(2004)
... to realities
Chemical Plants

Continuous Process Plant

Fieldbus

Field Configuration

Sensing and Acting

Process Configuration

Control Network

Enterprise Network

Data Storage

Process Operation

MIS

Business

Safety

Process Control

Chemical Plants
Computerised cars

- I/O Adaptor
- Driver Interface Adaptor
- Body Electronics Network Adaptor
- Gateway Body Adaptor
- Power Train Adaptor
- Suspension Adaptor
- Steering Manager Adaptor
- Brake Manager Adaptor
- I/O Adaptor
- Gateway Adaptor
- Driver Interface
Distribution Grids

SCADA SYSTEM

Remote Terminal Units

Data connections

Wide Area Network

Local Area Network

Regional Control Center 1

Regional Control Center n
"She's the very first Robopsychologist" -- Marvin Minsky, Father of Artificial Intelligence, MIT

Dr. Joanne Pransky
WORLD’S FIRST ROBOTIC PSYCHIATRIST®

... like Dr. Susan Calvin in “I, Robot”
Science and technology of computer-based control is facing an enormous challenge when increased levels of autonomy and resilience are required from the machines that are supporting our ambient control systems complexity is boosting.
In some sense, software intensive controllers are becoming too complex to be built by traditional software engineering methods.

Two main effects:

- Increase in size
- Decrease of dependability
Complexity forces change
Adaptation as solution

A possible path to the solution of this increasing complexity problem is adaptation.

Adaptation can be seen from many perspectives, e.g.

- adaptation during implementation
- runtime adaptation.
Examples

A paradigmatic example of the first type (construction) is **reusable component retargeting** to adapt it to a particular execution platform.

A paradigmatic example of the second type (run-time) is **fault-tolerant control**.
Can Consciousness help?

Can artificial consciousness be a solution to the performance/dependability problem?
The Obvious

What do we prefer?

- A conscious brain surgeon
- An unconscious brain surgeon

The reason: conscious real-time adaptation
The Obvious?

What do we prefer?

- A conscious *robotic* brain surgeon
- An unconscious *robotic* brain surgeon
It looks like it is better if machines are conscious

But:

Machines are not humans
Beware!

We must be aware of the differences between **sensing** and **perceiving**.

Perception is tinted with the colours of **phenomenology**, that seems to be a private, agent specific, issue (in biosystems).

We would need some form of alloperception for machines.
Motivations for AC

- **Artefacts like us**: consciousness, emotion and affect, experience, imagination, etc. (Robotics)
- **Studying natural systems** with computer laboratory models (Cognitive Science)
- **Proficient machines** (Intelligent Control)
The Focus of our work

Software intensive controllers for autonomous robust behaviour based on **self-awareness**

Qualia left for future endeavours
Autonomy Approach
An alternative approach is to move the responsibility for correct operation into the system itself.

That means moving the adaptation from the implementation phase into the runtime phase.

During runtime the system perceives changes and adapts to these changes to keep the mission assigned to it during the design phase.
Many Alternatives

Several alternatives are being explored based on the implementation of **architectural mechanisms** for self-organisation and/or self-repair.

These systems are built and started in a base state and they follow **adaptive life-cycles** based on the circumstances of the environment that surrounds the computing system (e.g. fault-tolerant systems, automatic learning, genetic programming or autonomic computing).
Autonomic Computing

IBM autonomic computing initiative:

"Autonomic computing is the ability of an IT infrastructure to adapt to change in accordance with business policies and objectives. Quite simply, it is about freeing IT professionals to focus on higher-value tasks by making technology work smarter, with business rules guiding systems to be self-configuring, self-healing, self-optimizing, and self-protecting."
Autonomic Element
Autonomic Systems
Engineering Models
A recent strategy to build complex embedded systems is **model-driven development** (MDD).

In the words of Jonathan Sprinkle:

[MDD is] “A design methodology used to create and evolve integrated, multiple-aspect models of computer-based systems using concepts, relations, and model composition principles to facilitate systems/software engineering analysis of the models and automatic synthesis of applications from the models.”
Deep and multiple system models are built and analysed using tools provided by the modelling environment. These models are later used to (automatically) generate the final system:

This means that the models must necessarily capture the semantic aspects of the final system and not just the structural properties.
MDD Approach

- Model Building Tools
- Behaviour Models
- Processing Models
- Hardware Models
- Model Analysis Tools
- System Generator
- Configuration Generator
- Runtime System

MODELS
GENERATORS
RUNTIME ENVIRONMENT
DESIGN ENVIRONMENT
Model-based Systems
Feedback control
Sample: RCS Model

Four core elements:

- Sensory Perception
- World Modelling
- Value Judgement
- Action Generation
Model-reference control

Reference Model

Adaptation Mechanism

Process

Controller

Reference model output

Control parameters

Command signal

Control signal

Process output

$y_m(t)$

$y(t)$
A truly adaptive system (a system with the autonomic properties) must necessarily do a semantic processing of the information it has about itself.

This is the type of reasoning that MDD tools help perform over their models if they are semantically precise enough.

Tools/Models that are being developed to be used by builders at the implementation phase can find their “autonomic” use by the system itself at the runtime phase.
Raising Self

Engineering models constitute the very self-model of the running system.

The (auto, allo) value system comes with the model.
Model Content

Model Exploitation

World
Self
Others
Reflective model-based

Classic adaptive control, or fault-tolerant control use **plant models** to adapt the control to new plant conditions. Fault-tolerant computing uses **computation models** to keep the computation ongoing.

**Reflective, model-based autonomous systems** will have deep model-based introspection capabilities that will let them achieve a high degree of adaptability and resilience.
Model Execution

- Models (of the world, of the self, of others) may be executed over virtual machines for different purposes:
  - Action calculation
  - Diagnosis and reconfiguration
  - What-if (imagination)
  - Retrodiction and causal analysis
Self-X functionality exploits models of the system itself in the performance of model-based action.
The Engineering Vision

- Break the barrier between the engineering process and the product
- Bridge the gap between construction and run-time
- Enable life-cycle adaptation

**Make the system self-aware** using software self-models + real-time reflection
Conclusions

- The rise of control system complexity claims for new technologies.
- Adaptation is a key issue. Self-adaptation is needed for many reasons (particularly dependability).
- Model technology is critical technology for complex controllers.
- Model-based control can be extended to the control system itself.
- Conscious controllers will exploit self models embedded in world models to maximise mission-level effectiveness.
A question remains:

What is it like to be a model-based reflective predictive controller?

“There may be something it is like to be such a self-model linked to such a world-model in a machine with a mission”
Thanks for the Consciousness