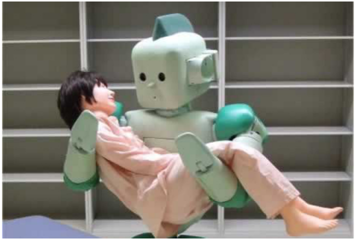
Verifiable Autonomy

Michael Fisher

University of Liverpool, 11th September 2015

Motivation: Autonomy Everywhere!



rtc.nagoya.riken.jp/RI-MAN

www.volvo.com

Motivation: Autonomous Systems Architectures

Many autonomous system architectures have been devised, e.g: *subsumption architectures*, *hybrid architectures*, ...

Increasingly popular approach $\longrightarrow$ *hybrid agent architectures*.

An *agent* captures the core concept of autonomy, in that it is *able to make its own decisions without human intervention*.

But: this still isn't enough, as we need to know *why*!

We need the concept of a "*rational agent*":

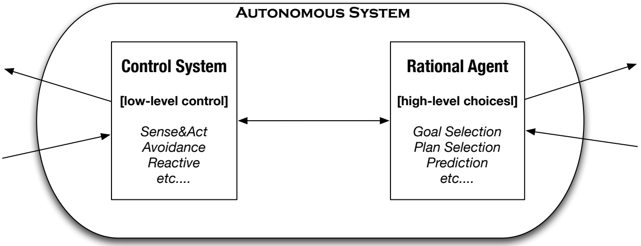
> a rational agent must have explicit *reasons* for making the choices it does, and should be able to explain these if needed

Motivation: Hybrid Agent Architectures

Requirement for *reasoned* decisions and explanations has led on to *hybrid agent architectures* combining:

1. *rational agent* for high-level autonomous decisions, and
2. traditional *control systems* for lower-level activities,

These have been shown to be easier to *understand*, *program*, *maintain* and, often, much more *flexible*.



Example: from Pilot to Rational Agent

Autopilot can essentially fly an aircraft

- keeping on a particular path,
- keeping flight level/steady under environmental conditions,
- planning route around obstacles, etc.

Human pilot makes high-level decisions, such as

- where to go to,
- when to change route,
- what to do in an emergency, etc.

Rational Agent now makes the decisions the pilot used to make.

RECAP: Programming Rational Agents

Programming languages for rational agents typically provide:

- a set of *beliefs* — information the agent has;
- a set of *goals* — motivations the agent has for doing something;
- a set of *rules/plans* — mechanisms for achieving goals;
- a set of *actions* — agent's external acts; and
- deliberation mechanisms for deciding between goals/plans.

Almost all of these languages are implemented on top of `Java`.

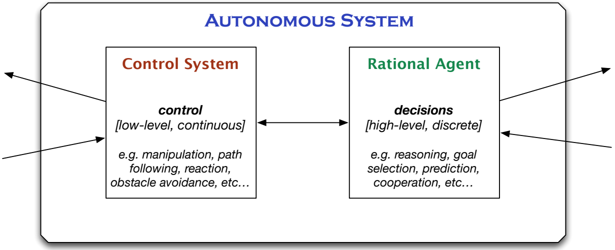
A typical agent rule/plan is:

```
Goal(eat) :  Belief(has_money), Belief(not has_food)
             <-  Goal(go_to_shop),
                   Action(buy_food),
                     Goal(go_home),
                       Action(eat),
                         +Belief(eaten).
```

What Shall we Verify?

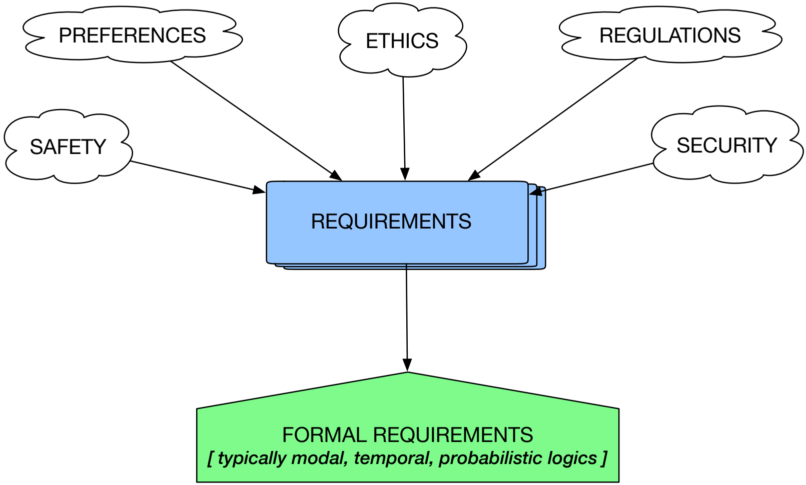
We want to verify the rational agent within the system's architecture.

Importantly, this allows us to verify the *decisions* the system makes, not its outcomes.



But: what logical properties shall we verify?

Formal Requirements



Example Logical Specification: Assisting Patients

In realistic scenarios, we will need to *combine* several logics.

> *If a patient is in danger, then the controller believes that there is a probability of 95% that, within 2 minutes, a helper robot will want to assist the patient.*

$B^{\geq 0.95}_{controller}$ . . . . *controller believes with 95% probability*

$\Diamond^{\leq 2}$ . . . . *within 2 minutes*

G_{helper} . . . . *helper robot has a goal*

$$in\_danger(patient) \Rightarrow B^{\geq 0.95}_{controller} \Diamond^{\leq 2} G_{helper}\, assist(patient)$$

Our Verification Approach

So, once we have

- an *autonomous system* based on rational agent(s), and
- a *logical requirement*, for example in modal/temporal logic,

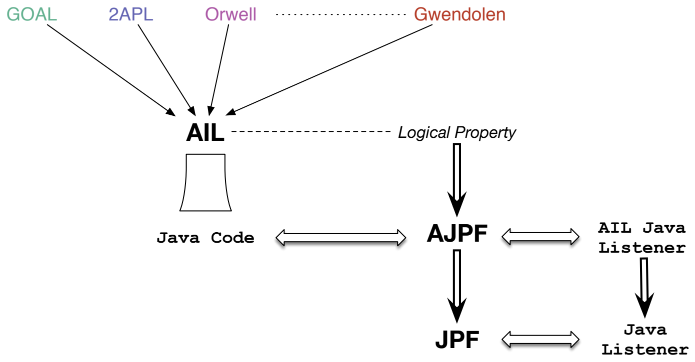
we have many options of how to carry out formal verification.

Approaches we can use include

- Proof: automated deduction in temporal/modal/probabilistic logics over a logical specification of the agent's behaviour,
- Traditional Model-Checking: assessing logical specifications over a model describing the agent's behaviour,
- Dynamic Fault Monitoring (aka Runtime Verification): watching for violations as the autonomous system executes,
- Program Model-Checking: assessing logical specifications against the *actual* agent code.

$\Rightarrow$ we are particularly concerned with this last one.

AJPF: Anatomy of an Agent Model Checker

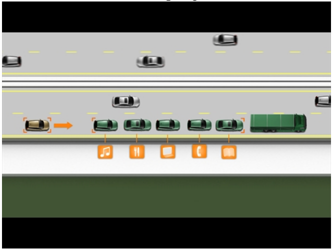


AJPF is essentially JPF2 with the theory of AIL *built in*.

The whole verification and programming system is called `MCAPL` and is freely available on Sourceforge:
`sourceforge.net/projects/mcapl`

Verification Example: Road Trains

www.sartre-project.eu:



Underlying control system manages distances between vehicles.
Rational agent makes decisions about joining/leaving, changing control systems, etc.

Verifying Rational Agent to ensure that convoy operates appropriately.

Ask Maryam/Owen for details

Verification Example: UAV Certification

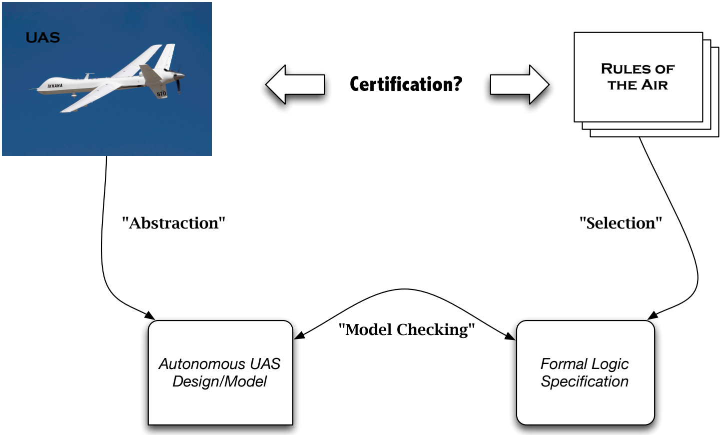
What's the core *difference* between a UAV and a manned aircraft?

Obviously: the UAV uses a "rational agent" instead of a pilot!

So, why can't we verify that the "agent" behaves just as a pilot would? i.e. is the agent *equivalent to* the pilot??

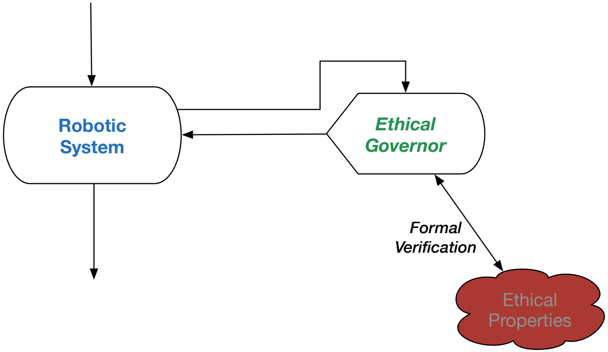
This is clearly *impossible*, but......

Our Approach



Ask Matt/Mike for details

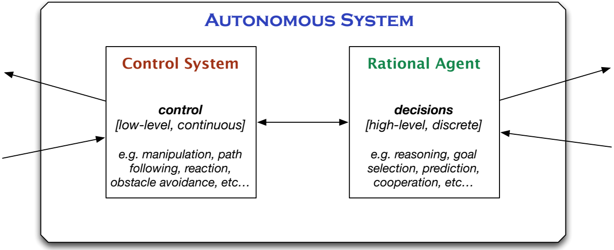
Verification Example: Ethical Decision-Making (1)



Ethical governor is essentially a rational agent, so verify this agent against ethical requirements/properties.

Ask Dieter/Louise for details

Verification Example: Ethical Decision-Making (2)



In unexpected situations, planners invoked and agent decides between options.

So verify the agent's decision-making approach against the appropriate ethical ordering.

Ask Louise for details

Concluding Remarks

Key new aspect in Autonomous Systems is that the system is able to *decide for itself* about the best course of action to take.

Rational Agent abstraction represents the core elements of this autonomous decision making:

- (uncertain) *beliefs* about its environment,
- *goals* it wishes wish to achieve and,
- *deliberation* strategies for deciding between options.

Clearly, *formal verification* is needed.

By verifying the rational agent, we verify not *what* system does, but what it *tries* to do and *why* it decided to try!

For this we need appropriate abstractions of the real control, sensing, etc, aspects.

Thanks to *many* people.....

The work described in this talk is due to *many* people.....

- Louise Dennis (Computer Science, Univ. Liverpool)
- Matt Webster (Computer Science, Univ. Liverpool)
- Clare Dixon (Computer Science, Univ. Liverpool)
- Maryam Kamali (Computer Science, Univ. Liverpool)
- Rafael Bordini (UFRGS, Brazil)
- Alexei Lisitsa (Computer Science, Univ. Liverpool)
- Sandor Veres (Engineering, Univ. Sheffield)
- Owen McAree (Engineering, Univ. Sheffield)
- Mike Jump (Engineering, Univ. Liverpool)
- Richard Stocker (NASA Ames Research Center, USA)
- Marija Slavkovik (Univ. Bergen, Norway)
- Alan Winfield (Bristol Robotics Lab)

- EPSRC, for funding many of these activities.

Sample Relevant Publications

- Dennis, Fisher, Slavkovik, Webster. Ethical Choice in Unforeseen Circumstances. In *Proc. TAROS 2013*.
- Dennis, Fisher, Webster. Verifying Autonomous Systems. *Communications of the ACM 56(9):84–93*, 2013
- Dennis, Fisher, Lincoln, Lisitsa, Veres. Practical Verification of Decision-Making in Agent-Based Autonomous Systems. To appear in *Journal of Automated Software Engineering*.
- Dennis, Fisher, Winfield. Towards Verifiably Ethical Robot Behaviour. Proc. First International Workshop on AI and Ethics. AAAI, 2015
- Dixon, Webster, Saunders, Fisher, Dautenhahn. Temporal Verification of a Robotic Assistant's Behaviours. In *Proc. TAROS 2014*.
- Lincoln, Veres, Dennis, Fisher, Lisitsa. Autonomous Asteroid Exploration by Rational Agents. *IEEE Computational Intelligence 8(4):25–38*, 2013.
- Webster, Cameron, Fisher, and Jump. Generating Certification Evidence for Autonomous Unmanned Aircraft Using Model Checking and Simulation. *J. Aerospace Information Systems 11(5):258–279*, 2014.