# Certifiable RAS Assurance using Integrated Formal Methods

Jim Woodcock

HISE, University of York

16th–17th January 2019

Huawei Research Symposium, Paris

# Collaborative Research

- Formal methods: long-term research agenda for HISE
  - Understand scientific basis for systems engineering
  - Transform current practice in dependable systems assurance
- Identify real strengths and weaknesses of formal methods
- Work with Mario Gleirscher & Simon Foster

- This talk: Assurance of Robotics and Autonomous Systems

- Opportunities for collaborative research
  1. Empirical studies of effectiveness of formal methods
  2. Integration of heterogenous formal methods
  3. Research into RAS assurance methodologies
  4. Technology transfer into industrial RAS assurance

- Overview of this talk
  1. Certifiable RAS assurance using integrated formal methods
  2. Directions for future collaborative research
  3. Expectations for useful research outcomes

# Background

Mounting anecdotal evidence on software-related risks.

- Need for rigorous discipline of software engineering
- 40 years of formal methods theory & practice
- Jim Woodcock et al.: Formal methods: Practice and experience. *ACM Comput. Surv.* 41(4): 19:1-19:36 (2009)
- Industrial applications show strengths and weaknesses
- Disciplines: requirements engineering, architectural design, test-driven development, program synthesis, testing

Working hypotheses.

1. Formalisms, techniques, tools increase engineering rigour.
2. Increased rigour can be helpful.
3. Better process achieves better outcomes cost-effectively.

# Background

- Nascent field: Robotic and Autonomous Systems (RAS)
- Increasing level of safety criticality
  - healthcare, autonomous vehicles, human-robotic interaction
- Regulatory acceptance requires assurance cases
- Comprehensible and indefeasible safety arguments
- Standards: IEC 61508 and DO-178C
- Laborious to create safety cases
- Complicated to maintain and evolve
- Must be rigorously checked by evaluators

# Mobile Autonomous Robots

- industrial robots: repetitive tasks in controlled setting
- current trend: from automatic to autonomous robots
- example: Huawei has first smartphone driven car
  - Porsche Panamera detects obstacles, understands surroundings
- open environments & human interaction: deeply safety critical


- challenge: how do we assure safety of autonomous robots?

# Robo∗ Projects at University of York

## Software Engineering for Mobile and Autonomous Robots

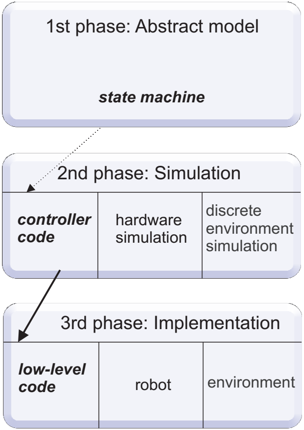
- RoboCalc, RoboTest, RoboSec, ...
- EPSRC, RAEng, Royal Society, ...

## Research programme objectives

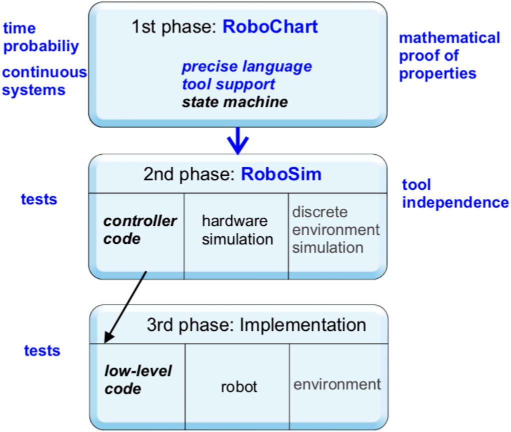
- Integrated modelling, simulation, programming for RAS
- Cover full development life-cycle
- Based on industry domain-specific notations
- Complemented with hardware & environment specifications
- Enriched with timed and probabilistic behaviours
- Simulation language linked to industry tools
- Powerful validation and verification techniques
- Combined notations, methods, automation, scalability

# Current Practice and RoboCalc

## Current practice




## RoboCalc vision



# Current Practice and RoboCalc

## Current practice

- Current practice based on standard state machines
- No formal semantics, specifies only controller
- No accepted paradigm for rigorous development
- State machine design guides simulation development
- No connection between design and simulation
- Reality gap between simulation/hardware/environment
- Impact on cost, maintainability, reliability

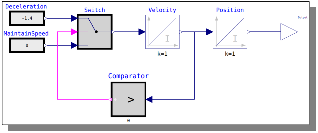
## RoboCalc Framework

- Transformative change through tools
- Guidance for practical development
- Pathway to certification for assurance
- Platform for sound tool development

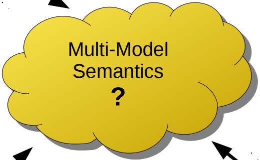
# Multi Model Semantics

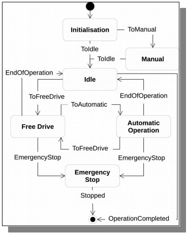
$$\dot{\psi} = \frac{(\omega_r - \omega_l)}{T_c}$$

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = V_c \begin{bmatrix}\cos(\psi)\\ \sin(\psi)\end{bmatrix}$$

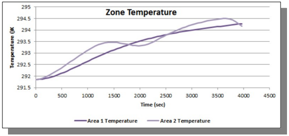



```c
#include <stdio.h>
#define MAX 10
int main()
{
    char array[MAX][MAX], c = 0;
    int d = 1, x = 0, i, j;

    do
        scanf("%s", array[x]);
    while (array[x++][0] != '0');

    {
        float* pf;
        int xx, *pi = (int*)&array[0][7];
        xx = ((*pi) & 0x41000000);
        pf = (float*)&xx;
        printf("%5.2f\n", *pf);
    }
```




```
class Controller

values

setPoint : real = 20;
tolerance : real = 0.5;

upperLimit : real = setPoint + tolerance;
lowerLimit : real = setPoint - tolerance;


instance variables
damperActuator : Actuator;
valveActuator : Actuator;
temperatureSensor   : Sensor;

fanStatus : real :=0.15 ;
valveStatus : real :=0 ;
```



# Challenge of Unification

## Science

- How do we classify and relate different languages?
  - UML, SysML, discrete control, Simulink, Modelica, real time, mobility, probability, . . .
- Are there core principles that unify them?
- How do we link and integrate their various formal semantics?
- How do we build modular & sound program analysis tools

## Engineering

- How do we apply integrated methods to autonomous robotics?
- What's the pay-off?

## Unifying Theories of Programming (UTP)

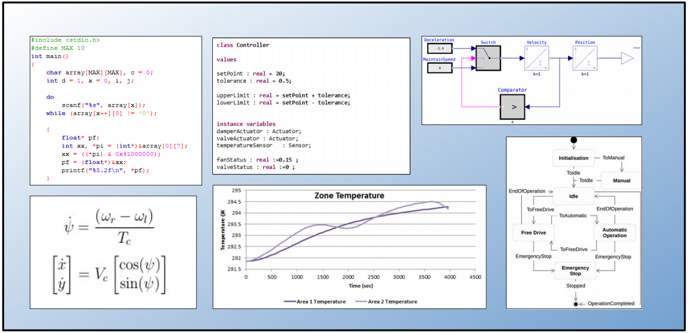
- Framework for defining and linking semantics
- Tony Hoare (Oxford/Microsoft), He Jifeng (ECNU/Shanghai)

# Overview of UTP

1. Meta-DSL for DSL semantics
   - Meta-logic: lenses & alphabets
   - Alphabetised relational calculus
   - Theories as complete lattices of relations
2. Core concepts
   - Design contracts: pre- and postconditions, refinement calculus
   - Verification calculi: Hoare, wp, operational semantics
3. Automation & compositionality
   - Mechanised semantics and dedicated theorem provers
   - Compositional formal definitions of complex languages
   - Individual parts available for reuse

UTP scales up to industrial-strength languages

# UTP Mechanised Semantic Tower




| | | | | |
|---|---|---|---|---|
| Graphical Notations | Object Orientation | Differential Equations | Concurrency | Contracts |

| | | | |
|---|---|---|---|
| Hybrid Systems | State | Real-time | Reactive Systems |

**Unifying Theories of Programming**

**Isabelle/HOL**

# Planned Contributions

1. Machine-checked assurance cases
   - Increase confidence in system dependability
   - Modularise arguments and evidence
   - Help maintenance and system evolution
2. Integrated formal methods
   - "Horses for courses"
   - Modern virtual prototyping
   - Hybrid verification tools and tool chains
   - Automated evidence gathering process
   - Safety "diff" on assurance case changes
3. Directions for foundational and empirical research
   - Methodology: iFMs for RAS assurance
   - Transfer to industrial RAS assurance
   - Empirical analysis of iFM/RAS applications

# Assuring Autonomous Systems: our Position

## Six propositions

1. Tools-based iFMs meet RAS safety-assurance challenge.
2. iFMs + modern verification automate evidence-gathering.
3. No stable path to assured autonomy without iFMs.
4. Depends on integrating FMs for different RAS aspects.
5. Semantic model integration necessary for iFM MDE.
6. Empirical research essential to determine if iFMs are effective.

# Research Objectives and Tasks

1. Empirical evaluation and technology transfer.
   - Evaluate assurance-case construction with iFMs
   - Debunk or justify arguments against the use of FMs
   - Transfer FM research to industrial assurance and certification
2. iFM foundations.
   - Integration and unification of FMs for RAS
   - Unified semantics for RAS assurance and tool integration
3. Evidence base.
   - Identify gap in assurance practice and assurance research
   - Understand how current RAS assurance practices fail
   - Suggest effective alternatives from assurance research
   - Understand how to validate assurance research

# Research Objectives and Tasks

4. Set directions for empirical FM research in RAS assurance.
   - Train FM researchers in applying empirical research methods
     - Formal methods champions need to measure costs
   - Understand information bias in scientific research
   - Increase the level of evidence of FM research
   - Avoid knowledge gaps
     - Between RAS practice and state of the art in assurance
     - Roadmap research directions against industrial requirements
5. Use appropriate research designs.
   - Engage RAS industry with recent iFM research
   - Goal-oriented interaction between practitioners and researchers
   - Summarise achievements in practical applications
   - Develop improved curricula for RAS assurance
   - Process improvement for assurance and certification
   - Guide vendors of FM tools to address assurance

# Research Questions addressing these Objectives

1. What is the true extent of computer-related accidents?
2. What about accidents in RAS domain?
3. To what extent do iFMs detect severe errors?
4. Does this improve on alternatives?
5. How do you measure iFM effectiveness?
6. How would COTS verification by iFMs pay off?
7. What are the obstacles to using iFMs in practice?
8. How do we know when they are overcome?
9. How do we unify FMs from different disciplines (iFMs)?
10. How do we combine formal and informal methods?
11. How can empirical research demonstrate iFMs in certifiable autonomy assurance?