Overview

1. History
2. Modelling and Simulation Concepts
3. Levels of Abstraction
4. Experimental Frame
5. Validation
6. Studying a mass-spring system
7. The Modelling and Simulation Process

Modelling and simulation: past

(1950–): Numerical simulations: numerical analysis, statistical analysis, simulation languages (CSSL, discrete-event world views).
focus: performance, accuracy

(1981–): Artificial Intelligence: model = knowledge representation
Use AI techniques in modelling, AI uses simulation (deep knowledge)
focus: knowledge

(1988–): Object-oriented modelling and simulation
focus: object orientation, later “agents”

Modelling and simulation: past, present, future

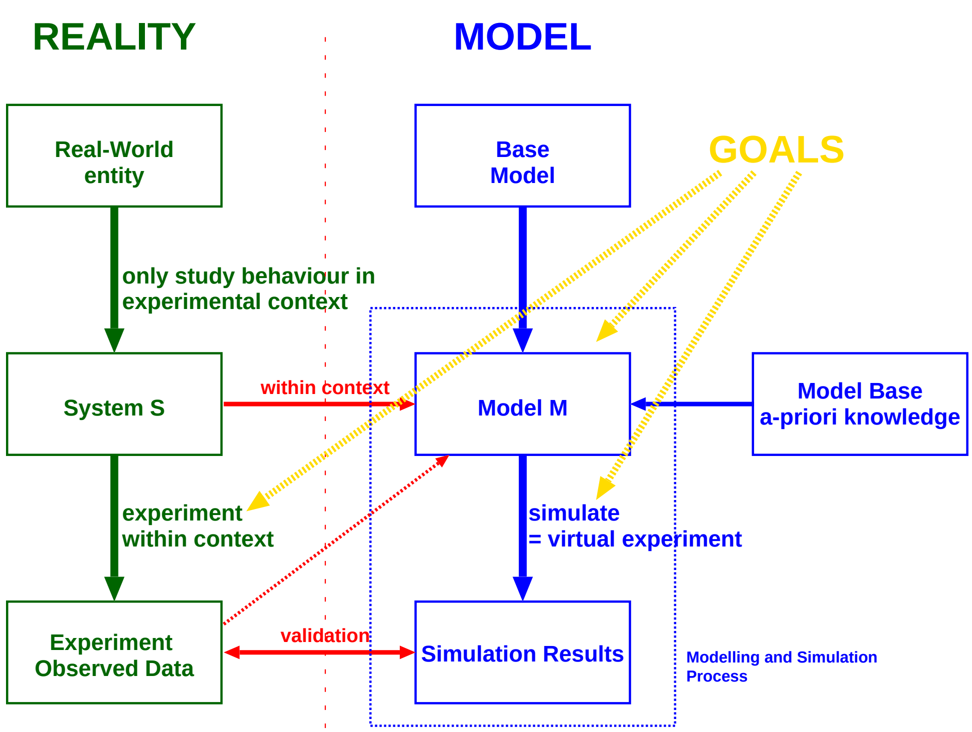
(1993–): ESPRIT Basic Research Working Group 8467 SiE

"Simulation for the Future: New Concepts, Tools and Applications"

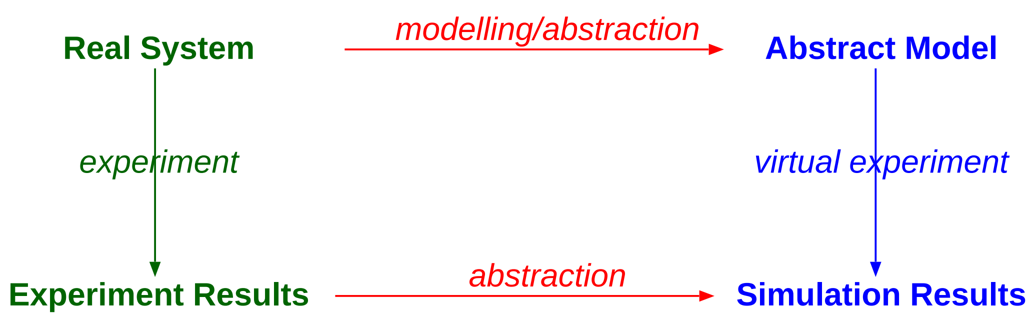
Industry: Gaz de France, PSA, DMI, VTT, Daimler, . . .

Academia: GMD First, DLR Oberphaffenhofen, CBL Leeds, . . .

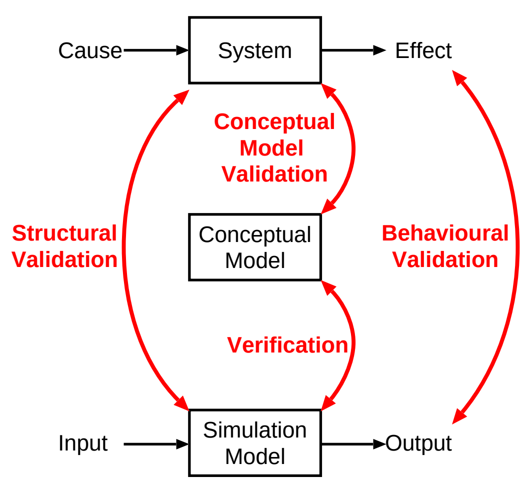
1. Do it right (optimally) the first time (market pressure)
2. Complex systems: **multi-formalism**
3. Hybrid: continuous-discrete, hardware/software
4. **Exchange** (between humans/tools) and **re-use** (validated model)
5. User focus: do not expect user to know details
   (software: glueing of components), need for **tools**



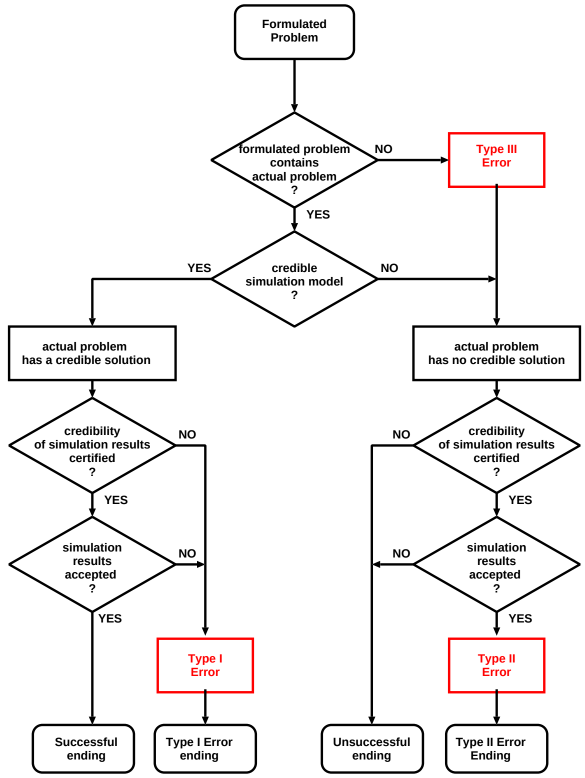
Behaviour Morphism



Verification and Validation



Popper: Falsification, Confidence

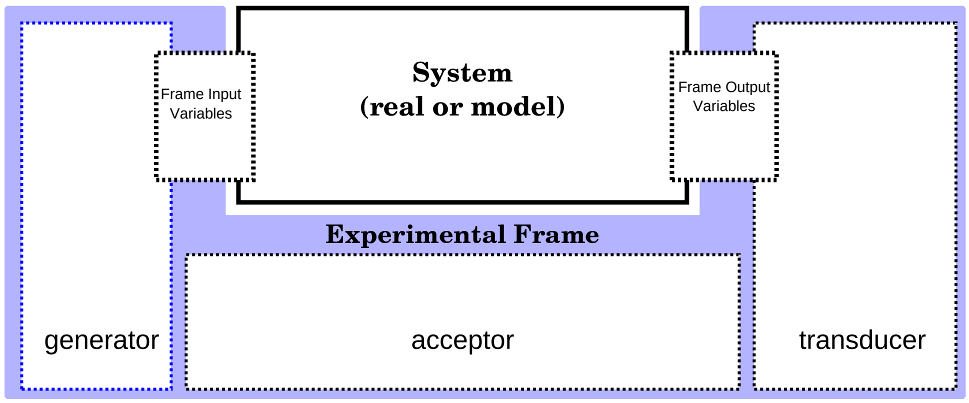


System, Base Model, Lumped Model

$$D_{BaseModel} \equiv D_{RealSystem}$$

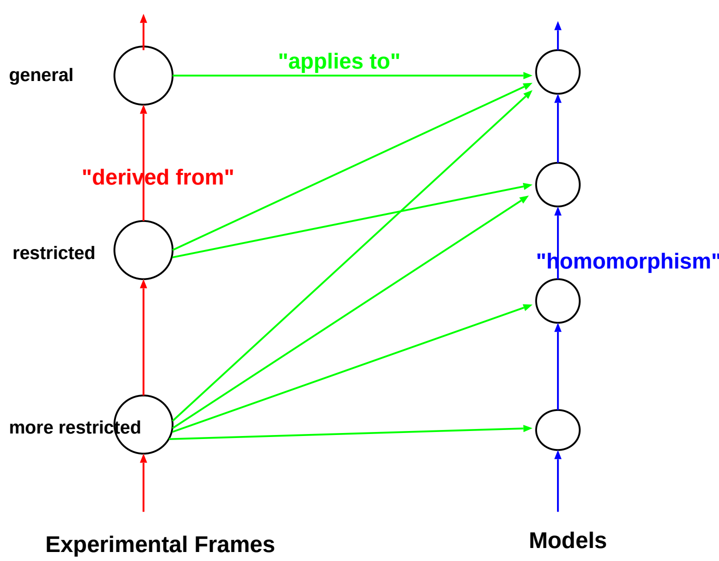
$$D_{LumpedModel} || E \equiv D_{RealSystem} || E$$

Experimental Frame Structure



$\sim$ Programming Language Types

Experimental Frame - Model Relationship



EF and Validity

Replicative Validity ($\equiv$: accuracy):

$$D_{LumpedModel}||E \equiv D_{BaseModel}||E$$

Predictive Validity:

$$F_{LumpedModel}||E \subseteq F_{BaseModel}||E$$

Structural Validity (morphism $\stackrel{\triangle}{=}$):

$$LumpedModel||E \stackrel{\triangle}{=} BaseModel||E$$

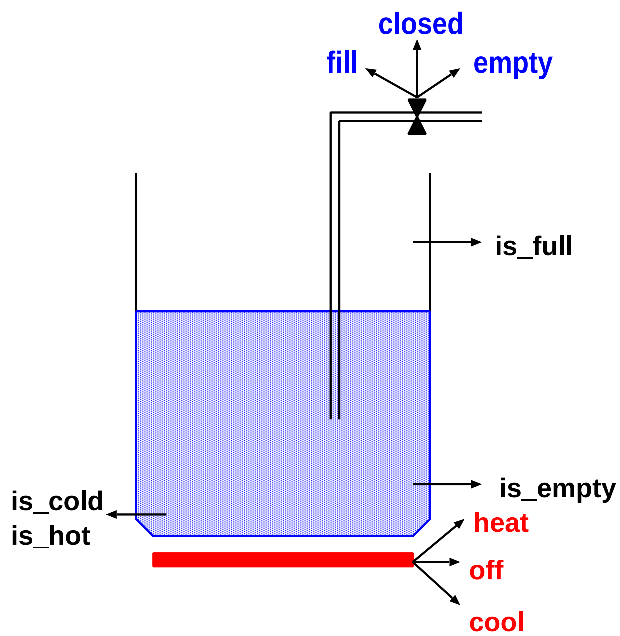
Simulator:

$$D_{Simulator} \equiv D_{LumpedModel}$$

Modelling Choices

1. System Boundaries: Experimental Frame (EF)
2. Level of Abstraction
3. Formalism(s)
4. Level of Accuracy

System under study: T, h controlled liquid



Detailed (continuous) view, ALG + ODE formalism

Inputs (discontinuous $\rightarrow$ hybrid model):

- Emptying, filling flow rate ϕ
- Rate of adding/removing heat W

Parameters:

- Cross-section surface of vessel A
- Specific heat of liquid c
- Density of liquid ρ

State variables:

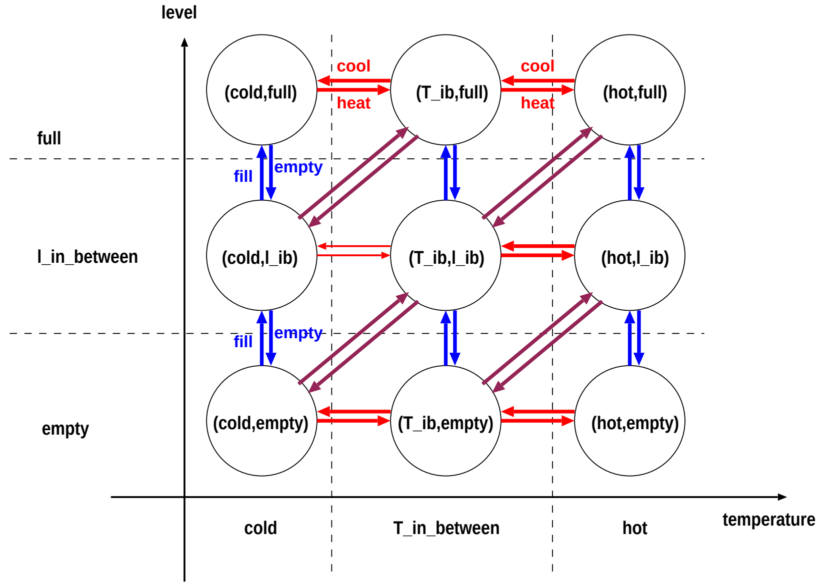
- Temperature T
- Level of liquid l

Outputs (sensors):

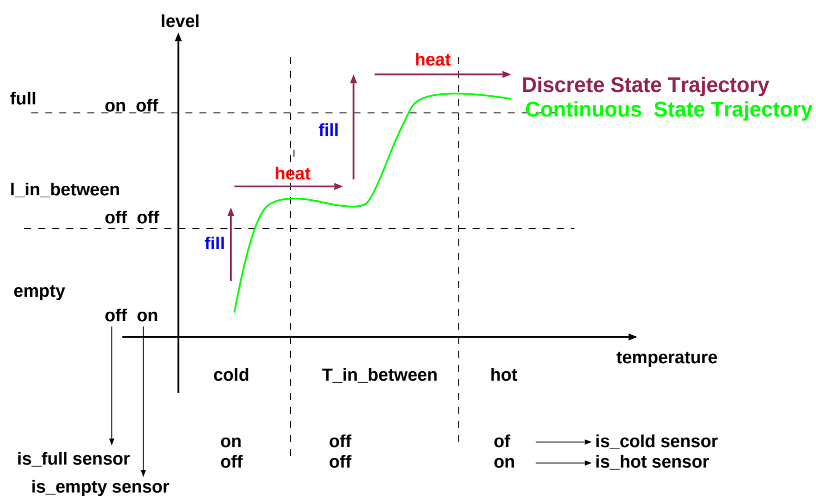
- $is\_low, is\_high, is\_cold, is\_hot$

$$\begin{cases} \frac{dT}{dt} &= \frac{1}{l}\left[\frac{W}{c\rho A} - \phi T\right] \\ \frac{dl}{dt} &= \phi \\ is\_low &= (l < l_{low}) \\ is\_high &= (l > l_{high}) \\ is\_cold &= (T < T_{cold}) \\ is\_hot &= (T > T_{hot}) \end{cases}$$

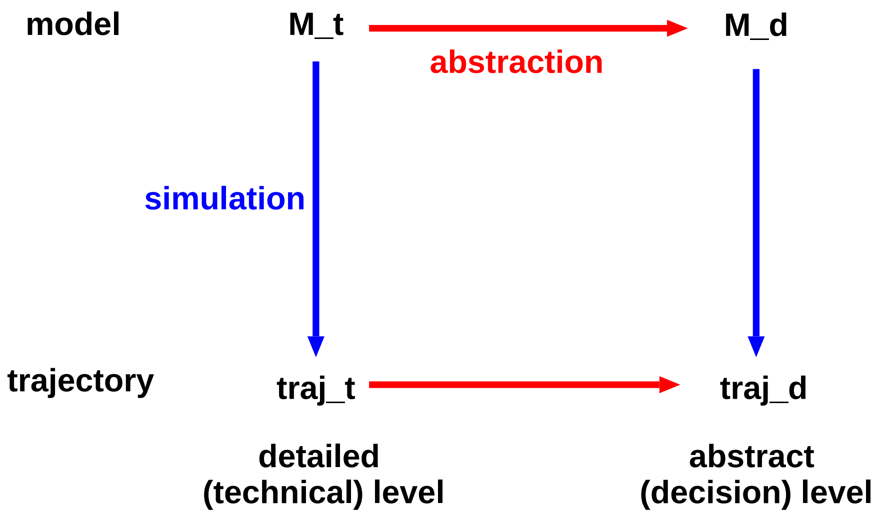
High-level (discrete) view, FSA formalism

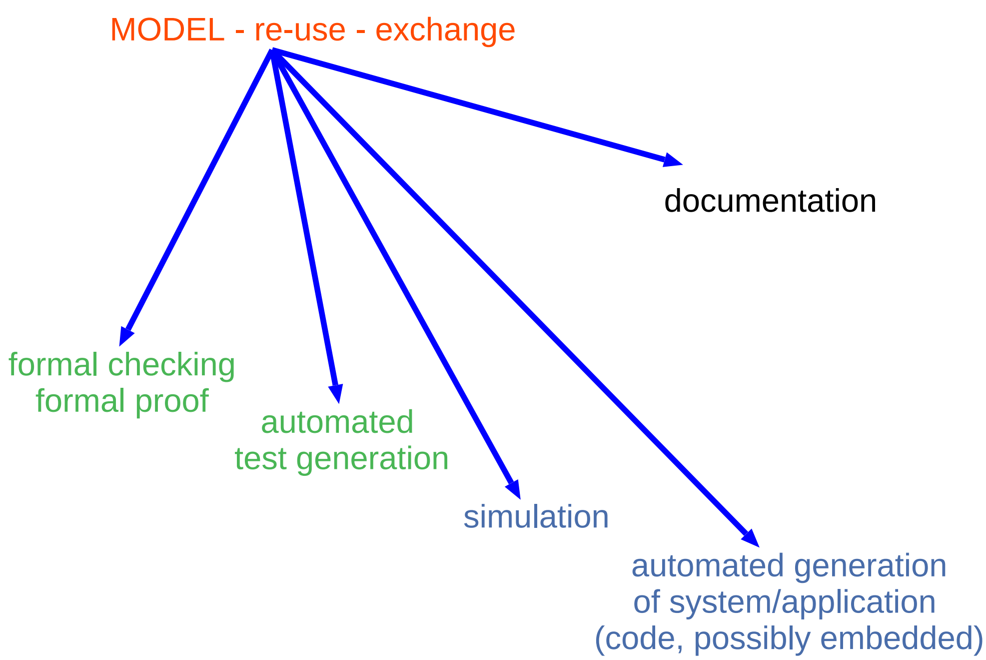


Levels of abstraction/views: trajectories

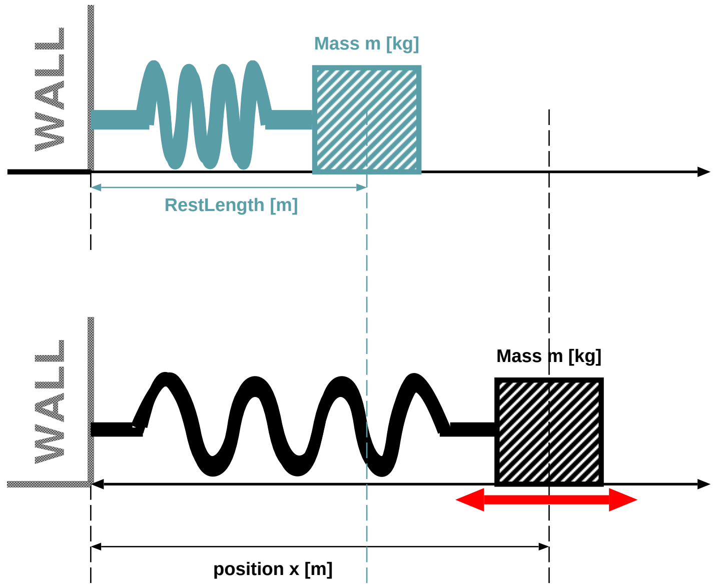


Levels of abstraction/views: morphism





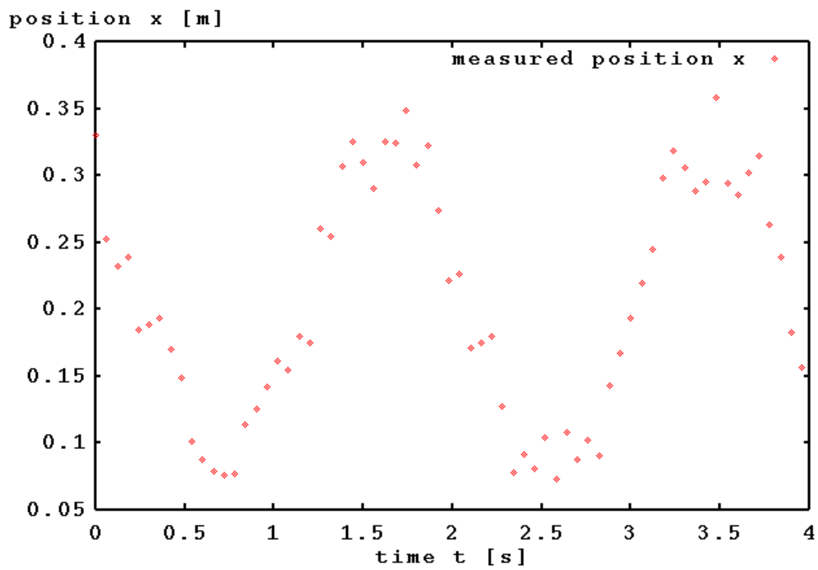
A Modelling and Simulation Exercise: the Mass-Spring system



Knowledge Sources

- A Priori Knowledge: Laws of Physics
- Goals, Intentions: Predict trajectory given Initial Conditions, “optimise” behaviour, . . .
  1. Analysis
  2. Design
  3. Control
- Measurement Data

Measured Data



Experimental Frame

- Room Temperature, Humidity, . . .
- Frictionless, Ideal Spring, . . .

Structure Characterisation

- Ideal Spring: Feature = maximum amplitude constant
- Spring with Damping: Feature = amplitute decreases

Building the model from a-priori knowledge

Newton's Law

$$F = M\frac{d^2\Delta x}{dt}$$

Ideal Spring

$$F = -K\Delta x$$

$$\downarrow$$

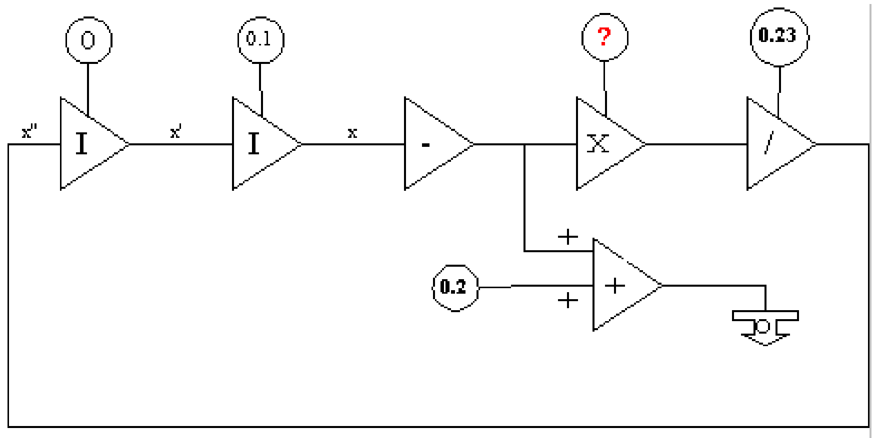
$$\frac{d^2\Delta x}{dt^2} = -\frac{K}{M}\Delta x$$


```
CLASS Spring "Ideal Spring": DAEmodel :=
{
 OBJ F_left: ForceTerminal,
 OBJ F_right: ForceTerminal,

 OBJ RestLength: LengthParameter,
 OBJ SpringConstant: SCParameter,

 OBJ x: LengthState,
 OBJ v: SpeedState,

 F_left - F_right = - SpringConstant * (x - RestLength),
 DERIV([ x, [t,] ]) = v,

 EF_assert( x - RestLenght < RestLength/100),
},
```

From Model to Simulation

Block-diagrams

analog computers, Continuous System Modelling Program (CSMP)



Time-slicing simulator pseudo-code

```
Main program:

 FOREACH block IN system
  IF block is integrator
   initialise block's output with initial condition (IC)
  ELSE
   initialise output with 0

 READ system network (graph) structure

 READ integrator configuration info
  step_size
  communication_interval

 READ experiment info
  end_time
```

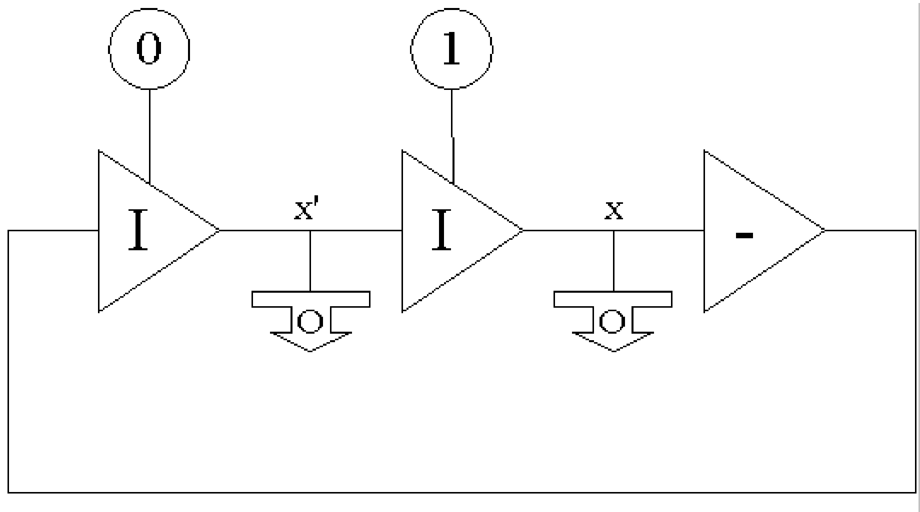
Time-slicing simulator pseudo-code (ctd.)

```
Simulation Kernel Loop:
 WHILE (NOT End_of_simulation) DO
  Update_blocks
  Output time and state variables

Update_blocks:
 FOREACH block IN system
  given current block inputs
  (get input from output of influencer)
  Calculate_block_output for block
 increment current_time with stepsize

End_of_simulation:
 termination condition such as
  current_time >= end_time
  condition(state_values) == TRUE
```



```
Calculate_block_output: ([...] means optional)

  WeightedSum
    W, block_number, P1, e1[, P2, e2[, P3, e3]] ; --->
       output= SUM_i(Pi*ei)
  Summer
    +, block_number, P1, e1[, P2, e2[, P3, e3]] ; --->
       output = SUM_i(sign(Pi)*ei)
       (only the sign of Pi is used)
  Integrator
    I, block_number, IC, e1 ; --->
       output= previous_output +  step_size*e1
       (simple fixed-step Euler integration)
  Minus (Sign Inverter)
    -, block_number, e1 ; --->
       output= -e1.
  Multiplier
    X, block_number, e1, e2 ; --->
       output= e1*e2.
  Divider
    /, block_number, e1, e2 ; --->
       output= e1/e2.
  Constant
    K, block_number, P1 ; --->
       output= P1.
  Output
    O, block_number, e1 ; --->
       output= e1.
       (As a side-effect, the (time, e1) tuple is put
        on the output stream at every communication point).
```

Time Slicing: Circle Test

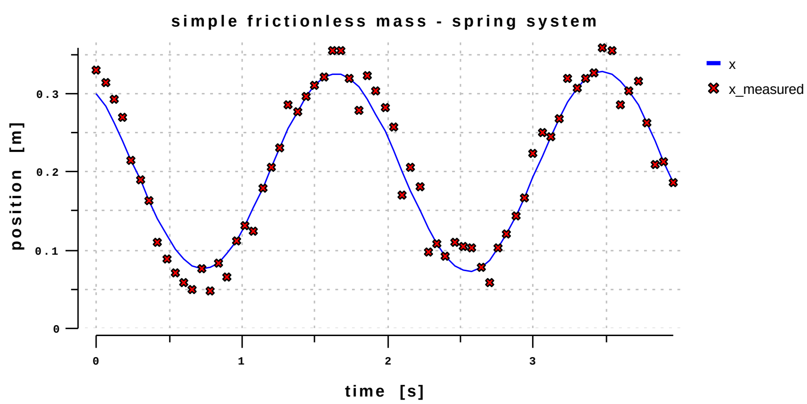



```
; Circle Test for
; CSMP-style Time Slicing Simulator

$endtime = 100;
$timestep = ?;
$comminterval = 1.5;

I, 1, 0, 3        ; x′ is integral of x′′, IC=0
I, 2, 1, 1        ; x  is integral of x′,  IC=1
-, 3, 2           ; -x
O, 4, 1           ; output x′
O, 5, 2           ; output x
```

Results Analysis

- Accuracy (numerical)
- Model Parameters
- Model Structure
- assignment . . .

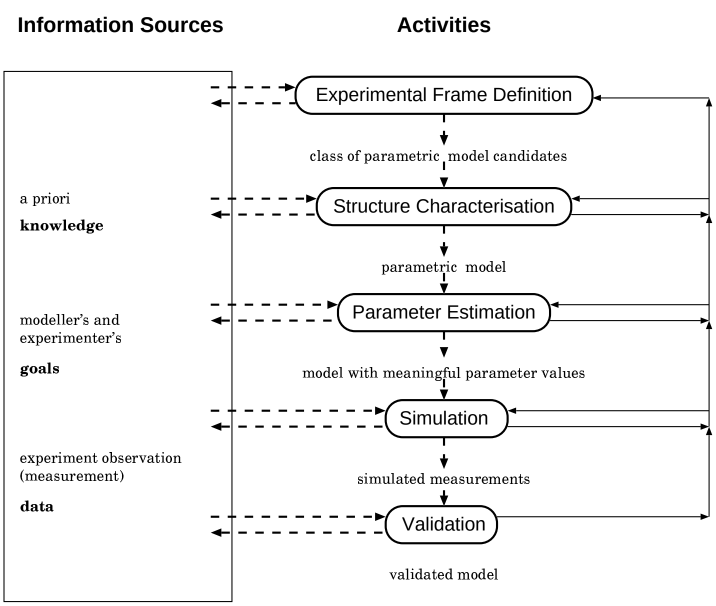
Model Calibration: Parameter Fit

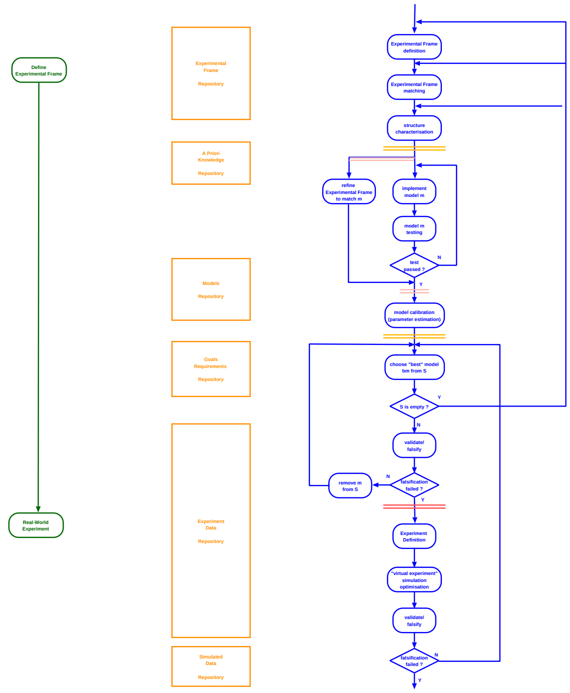


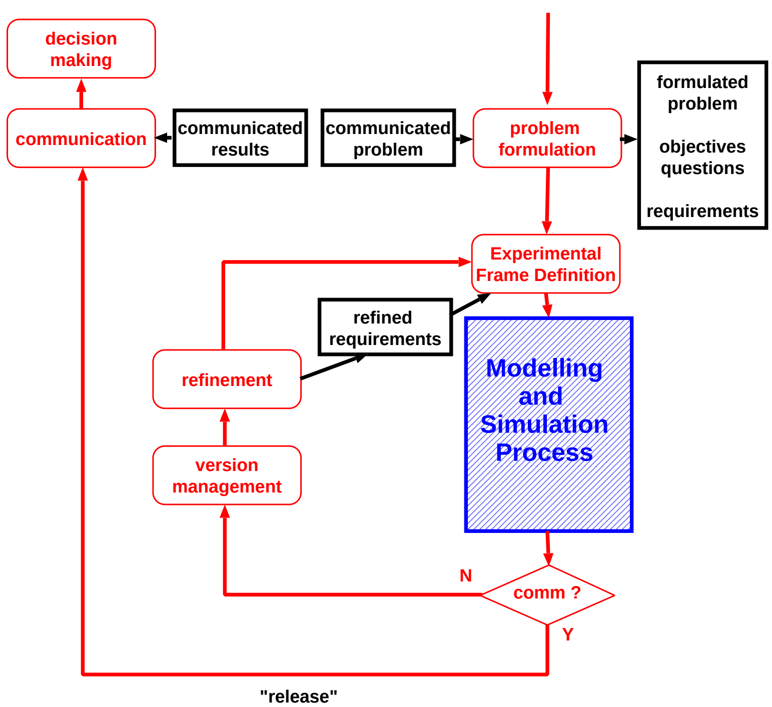
From Here On . . .

- Virtual Experiments: simulation, optimisation, what-if, . . .
- Validation/Falsification

Modelling and Simulation *Process*




Define Experimental Frame
Real-World Experiment
Experimental Frame Repository
A Priori Knowledge Repository
Models Repository
Goals Requirements Repository
Experiment Data Repository
Simulated Data Repository
Experimental Frame definition
Experimental Frame matching
structure characterisation
refine Experimental Frame to match m
implement model m
model m testing
test passed ?
N
Y
model calibration (parameter estimation)
choose "best" model bm from S
S is empty ?
Y
N
validate/ falsify
remove m from S
falsification failed ?
N
Y
Experiment Definition
"virtual experiment" simulation optimisation
validate/ falsify
falsification failed ?
N
Y


decision making
communication
communicated results
communicated problem
problem formulation
formulated problem
objectives questions
requirements
Experimental Frame Definition
refined requirements
refinement
Modelling and Simulation Process
version management
N
comm ?
Y
"release"