some context for Adrian Pop’s thesis

Integrated Model-Driven Development Environments for Equation-Based Object-Oriented Languages

Hans Vangheluwe

Modelling, Simulation and Design Lab (MSDL)
School of Computer Science, McGill University, Montréal, Canada
Overview

1. Why/When Simulation of Complex Systems?
2. Modelling and Simulation Concepts
3. Modelling Languages Evolution
4. Desirable Features
5. Domain-specific Modelling and Language Engineering
Simulation ... when too costly/dangerous

analysis $\leftrightarrow$ design
Simulation . . . real experiment not ethical

training, physical simulation
Simulation . . . evaluate alternatives
Simulation . . . “Do it Right the First Time”
Modelling/Simulation . . . and code/app Synthesis
Complex Systems: complexity due to ... 

large number of components
Complex Systems: complexity due to . . .


**heterogeneity** of components
Complex Systems: complexity due to . . .

**non-compositionality** of networks ("emergent" behaviour)
Modelling/Simulating Complex Systems . . .

- at the most appropriate **level of abstraction**

- using the most appropriate **formalism(s)**
  Ordinary Differential Equations, Petri Nets, Bond Graphs, Statecharts, Forrester System Dynamics, CSP, Queueing Networks, . . .
Defining “simulation”

**REALITY**

Real-World entity

only study behaviour in experimental context

System S

experiment within context

Experiment Observed Data

**MODEL**

Base Model

Model M

within context

Simulate = virtual experiment

Simulation Results

**GOALS**

Model Base a-priori knowledge

Modelling and Simulation Process

Hans Vangheluwe hv@cs.mcgill.ca

Modelling and Simulation: System Specification
Modelling Languages: Simulation Code Library

```c
#include "rundeck_opts.h"

MODULE STATIC_OCEAN

@sum STATIC_OCEAN contains the ocean subroutines common to all Q-flux

@+ and fixed SST runs

@auth Original Development Team

@rev 1.0 (Q-Flux ocean)

@cont OSTRUC, OCLM, init_OCEAN, daily_OCEAN, DIAGCO

@+ PRECIP_OC_OCEANS

USE CONSTANT, only:

@+ lh, rhi, h0, swh, shi, by2, bysh

USE MODEL_COM, only:

@+ lm, im, m0cean, fland, fearth, flice

@+ lyea1, ltime, lmon, ldate, lday, lyear, lmp, J0end0FM, J0m10FM

@+ ltimei, kocean, itocean, itoice

USE GEOM

USE PBLCOM, only:

@+ npbl, uab, vabi, tab, qabi, eabi, cm=>cmgs, ch=>chgs

@+ cqi=>cqq, lpb, rougi

USE SEAICE, only:

@+ xsi, acel1, z1i, ac2olm, z2olm, xsi10, tfrez, fleadoc

@+ lni

USE SEAICE_COM, only:

@+ rsi, msi, xsi, snowi, ssi

#ifdef TRACERS_WATER

@+ tsi, trs10, ntr
@endef

USE LANDICE_COM, only:

@+ snowi, tlandi

USE FLUXES, only:

@+ gtemp, ssh, fws1m, mlhc

USE DIAGCOM, only:

@+ aij, areg, jreg, ij_smfx, aj, j_impf, j_impfm,

@+ j2_impf, j2_impfm, j_smff1

IMPLICIT NONE

SAVE logical :: off_line, false,

@dbparam ocn_cycl determines whether prescribed ocean data repeat

@+ after 1 year - 0:no 1:yes 2:seaice repeats, sst does not

integer :: ocn_cycl = 1

@var TOCEAN temperature of the ocean (C)

REAL*8, DIMENSION(3,IM,JM) :: TOCEAN

@var SST0 default sea surface salinity (psu)

REAL*8 :: SST0=34.70

@var OTA, OTB ocean heat transport coefficients

REAL*8, DIMENSION(IM,JM,I) :: OTA, OTB

REAL*8, DIMENSION(IM,JM) :: OTB

@var SINANG, COSANG Fourier coefficients for heat transports

REAL*8 :: SINANG, SN2ANG, SN3ANG, SN4ANG,

@+ COSANG, CS2ANG, CS3ANG, CS4ANG

@var Z10 ocean mixed layer depth

REAL*8, DIMENSION(IM,JM) :: Z10

@var Z1OLD previous ocean mixed layer depth

REAL*8, DIMENSION(IM,JM) :: Z1OLD

@var Z120 annual maximum ocean mixed layer depth

REAL*8, DIMENSION(IM,JM) :: Z120
```
Modelling Languages: CSSL

```csl
program limit cycle
DERIVATIVE

!-----define all the preset variables
CONSTANT k = 0.2, tstop = 10.0
CONSTANT lp = 1.0

!-----communication interval
CINTERVAL cint = 0.2

!-----give name to radius value
sq = SQRT(x**2 + y**2)

!-----common factor
kk = (lp - sq**2)/sq

!-----limit cycle equations
CONSTANT xz = 0.5, yz = 1.0
x = INTEG(y + kk*x, xz)
y = INTEG(-x + y*kk, yz)

!-----define stopping condition
TERMT(t .GE. tstop, 'Time Limit')

END ! of derivative
end
```
Modelling Languages: Causal Block Diagram
Non-Causal Modelling Languages
Modelica: Electrical Resistor

model Resistor "Ideal linear electrical resistor"
   extends OnePort;
   parameter Resistance R=1 "Resistance";
   equation
       R*i = v;
end Resistor;
Modelica: Electrical Network

model circuit
    Resistor R1(R=10);
    Capacitor C(C=0.01);
    Resistor R2(R=100);
    Inductor L(L=0.1);
    VsourceAC AC;
    Ground G;

equation
    connect(AC.p, R1.p);
    connect(R1.n, C.p);
    connect(C.n, AC.n);
    connect(R1.p, R2.p);
    connect(R2.n, L.p);
    connect(L.n, C.n);
    connect(AC.n, G.p);
end circuit;
Modelling Languages: EcosimPro
Modelling Languages: Visual Modelling
Visual Modelling Environments: gProms
Quality Criteria for (Modelling) Languages

- well-defined syntax and semantics
- modularity
- library mechanism (and standard libraries)
- appropriate abstraction level for particular task
  (e.g., virtual experimentation with physical systems)
Were missing for Modelica:

- Tools for debugging
- Language features Beyond Equations
Beyond Equations Wishlist

- Tools for going “under the hood”
- Multi-formalism modelling
- Domain-Specific (visual) modelling
- Ability to bootstrap
Meta-Modelica and meta-programming

eat your own dogfood!
Multi-formalism / Heterogeneous MoC (Ptolemy)

solution:

- co-simulation
- formalism transformation
DSM: Waste Water Treatment Plants (WWTPs)

NATO’s Sarajevo WWTP

www.nato.int/sfor/cimic/env-pro/waterpla.htm
DS(V)M Environment

www.hemmis.com/products/west/

Hans Vangheluwe
hv@cs.mcgill.ca
Modelling and Simulation: System Specification
Why DS(V)M?
(as opposed to General Purpose modelling)

• match the user’s mental model of the problem domain

• maximally constrain the user (to the problem at hand)
  ⇒ easier to learn
  ⇒ avoid errors

• separate domain-expert’s work
  from analysis/ transformation expert’s work

Anecdotal evidence of 5 to 10 times speedup

Steven Kelly and Juha-Pekka Tolvanen.

Warning: A Fool with a Tool . . .