Heterogeneous Programming and Modeling of Cyber-Physical Systems

Smart Programming Day, SICS Software Week

November 29, 2016

David Broman

Associate Professor, KTH Royal Institute of Technology

Collaborators: Jeremey Siek (Modelyze) and Saranya Natarajan (Timed C)

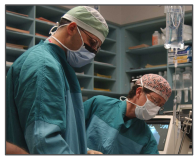
Examples of application areas

Automotive
(systems of systems)

Industrial
Automation

Aircraft
(traditional or
autonomous)

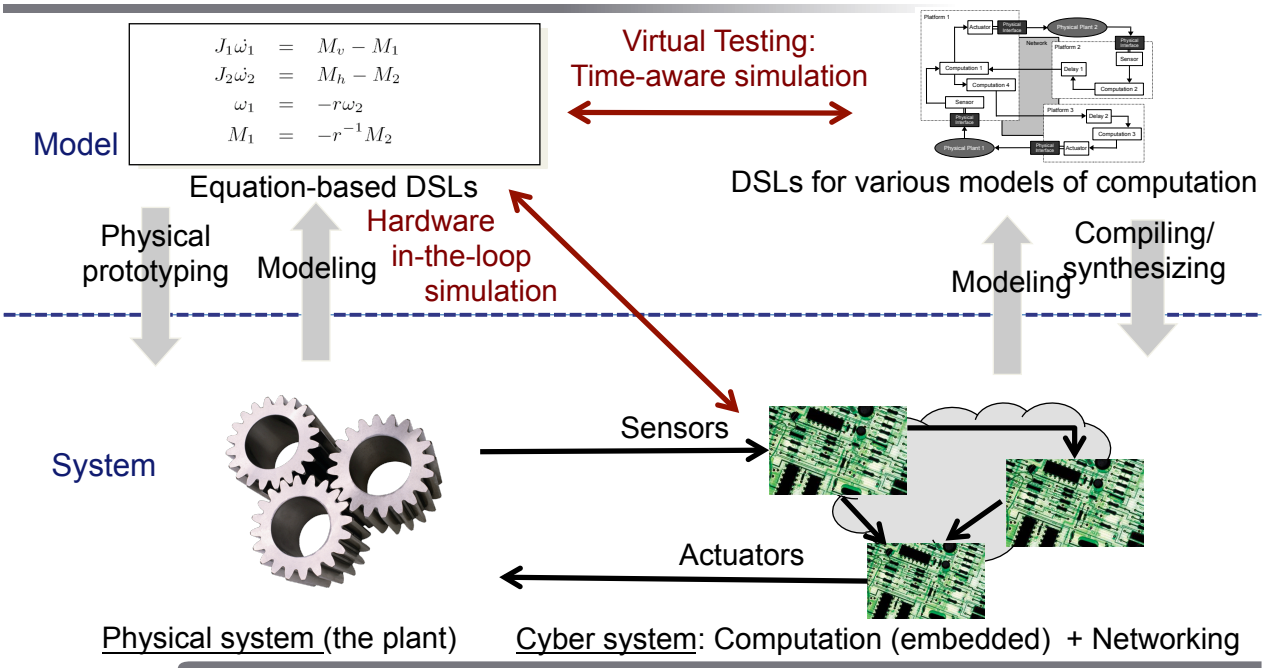
Satellites



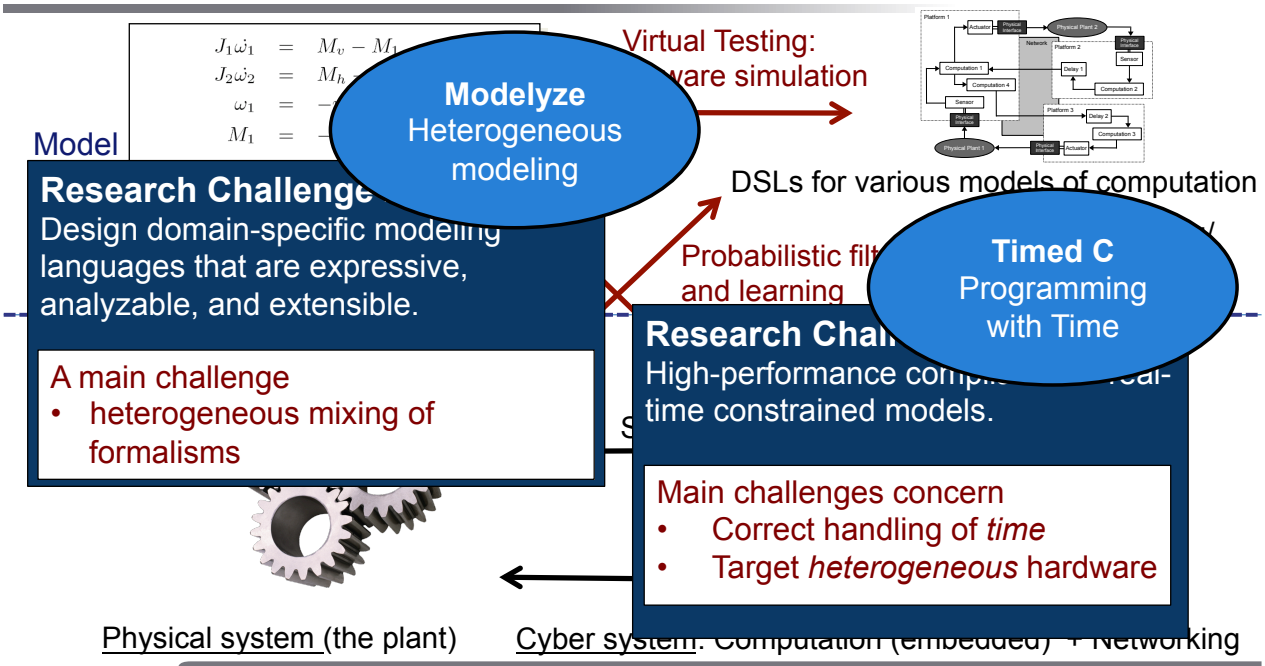
Medical
Equipment

Cyber-Physical Systems (CPS)

Heterogeneous Model-Based CPS Design



Heterogeneous Model-Based CPS Design



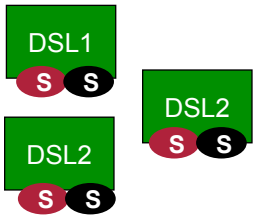
Our approach to heterogeneous modeling: Embedded Domain-Specific Languages (DSLs)

modelyze

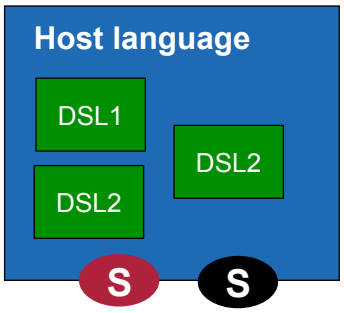
(MODEL and analYZE)

Small, simple, host language for **embedding domain-specific languages** (DSL) of different models of computation (MoC)

External DSL



Embedded DSL



Open source:
www.modelyze.org

Modelyze: A Gradually typed functional language.
Is a general purpose language, but designed for embedding.

A DSL for mathematical modeling embedded in Modelyze

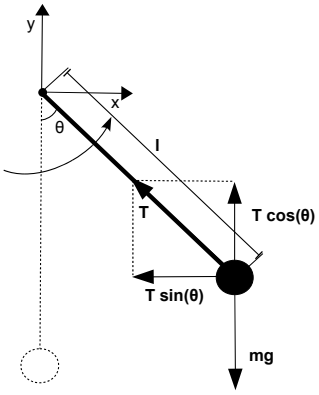
Equations and initial values are defined declaratively, just as the mathematical equations

```
def Pendulum(m:Real,l:Real,angle:Real) = {
    def x,y,T:Real;
    init x (l*sin(angle));
    init y (-l*cos(angle));

    T*x/l = m*x'';
    -T*y/l - m*g = m*y'';
    x^2. + y^2. = l^2.;
}
```

Differential-Algebraic Equations (DAEs).

y" means second order derivative



$$-T \cdot \frac{x}{l} = m\ddot{x}$$

$$-T \cdot \frac{y}{l} - mg = m\ddot{y}$$

$$x^2 + y^2 = l^2$$

$$x(0) = l\sin(\theta_s)$$

$$y(0) = -l\cos(\theta_s)$$

Declarative Mathematical Model

Which parts are part of the host language (Modelyze)?

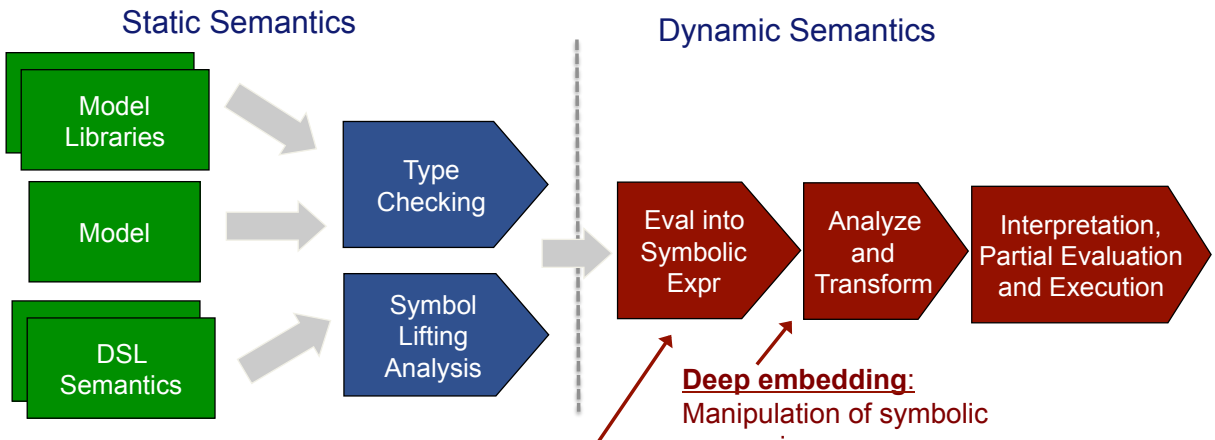
Unknowns are internally represented as typed symbols

```
def Pendulum(m:Real,l:Real,angle:Real) = {
    def x,y,T:Real;
    init x (l*sin(angle));
    init y (-l*cos(angle));

    -T*x/l = m*x'';
    -T*y/l - m*g = m*y'';
    x^2. + y^2. = l^2.;
}
```

Syntax and semantics for differential equations are **embedded** into the host language Modelyze.

Embedding and Execution Process



Deep embedding:
Manipulation of symbolic expressions

Shallow embedding:
Types, functions, etc. of the host language are used directly as part of the DSL

Embedding and Execution Process

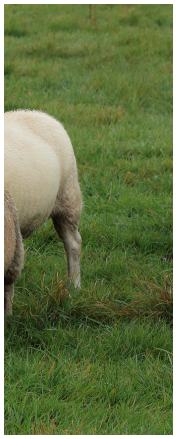
SHallow and d**EEP**

Let us mispronounce this a bit…

Shallow embedding:
Types, functions, etc. of the host language are used directly as part of the DSL

Deep embedding:
Manipulation of symbolic expressions

Embedding and Execution Process



Cheap embedding
The aim of combining the convenience of shallow embedding with the power of deep embedding.

Other names for combining shallow and deep embedding:
neritic (Augustsson, 2012) and Yin-Yang in Scala (Jovanovic et al., 2014)

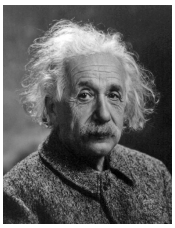
SHallow and d**EEP**

Let us mispronounce this a bit…

Shallow embedding:
Types, functions, etc. of the host language are used directly as part of the DSL

Deep embedding:
Manipulation of symbolic expressions

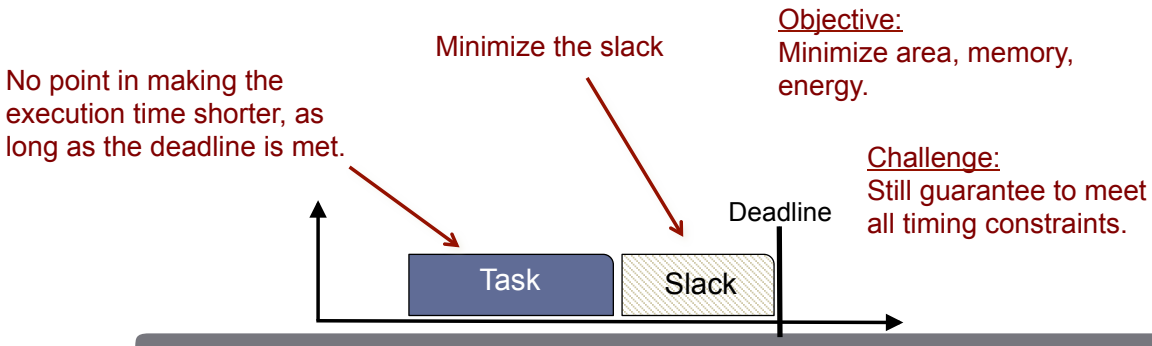
What is our goal?



"Everything should be made as simple as possible, but not simpler"

attributed to Albert Einstein

Execution time should be as short as possible, but not shorter

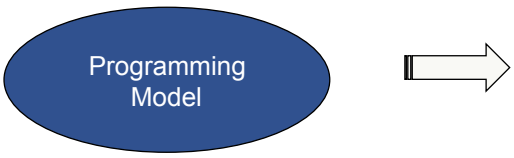


Programming Model and Time

Timing is not part of the software semantics

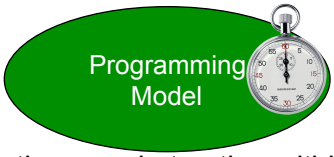
The correctness of programs is not related to execution time.

Traditional Approach

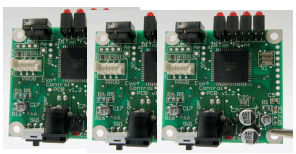


Timing Dependent on the Hardware Platform

Our Objective

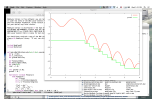


Make time an abstraction within the programming model



Enable timing **portability**, where timing requirements are verified by the compiler.

Compilation and Analysis



Timed and/or synchronous modeling languages (Simulink, Modelica, Ptides, SCADE etc.)

Timed C programs

May act as an **intermediate language**.

Source-to-source compilation

C Compiler

Distributed real-time high performance platforms (RTOS, POSIX API, Linux, Windows)

Timing Analysis

C Compiler

Distributed embedded microcontrollers (bare metal implementations)

Conclusions

Some key points:

modelyze

Modelyze is an ongoing project for embedding heterogeneous domain-specific modeling languages.

Timed C is an ongoing project where we incorporate real-time into low level languages.

For more info, visit our group page:
Model-based Computing Systems (MCS)
http://www.kth.se/ict/mcs

Thanks for listening!