TECHNISCHE UNIVERSITÄT DRESDEN

4th MODPROD 2010

Chair of Construction Machines and Conveying Technology

Using Modelica for Interactive Simulations of Technical Systems in a Virtual Reality Environment

J. Frenkel • C. Schubert
Prof. Dr.-Ing. habil. G. Kunze

Outline

1. Introduction
2. Simulation Framework
   SARTURIS & Modelica
3. Multibody Library
   PyMbs
4. Demonstration
5. Conclusion

TECHNISCHE UNIVERSITÄT DRESDEN

1. Introduction

Current Topics of Research at our Institute

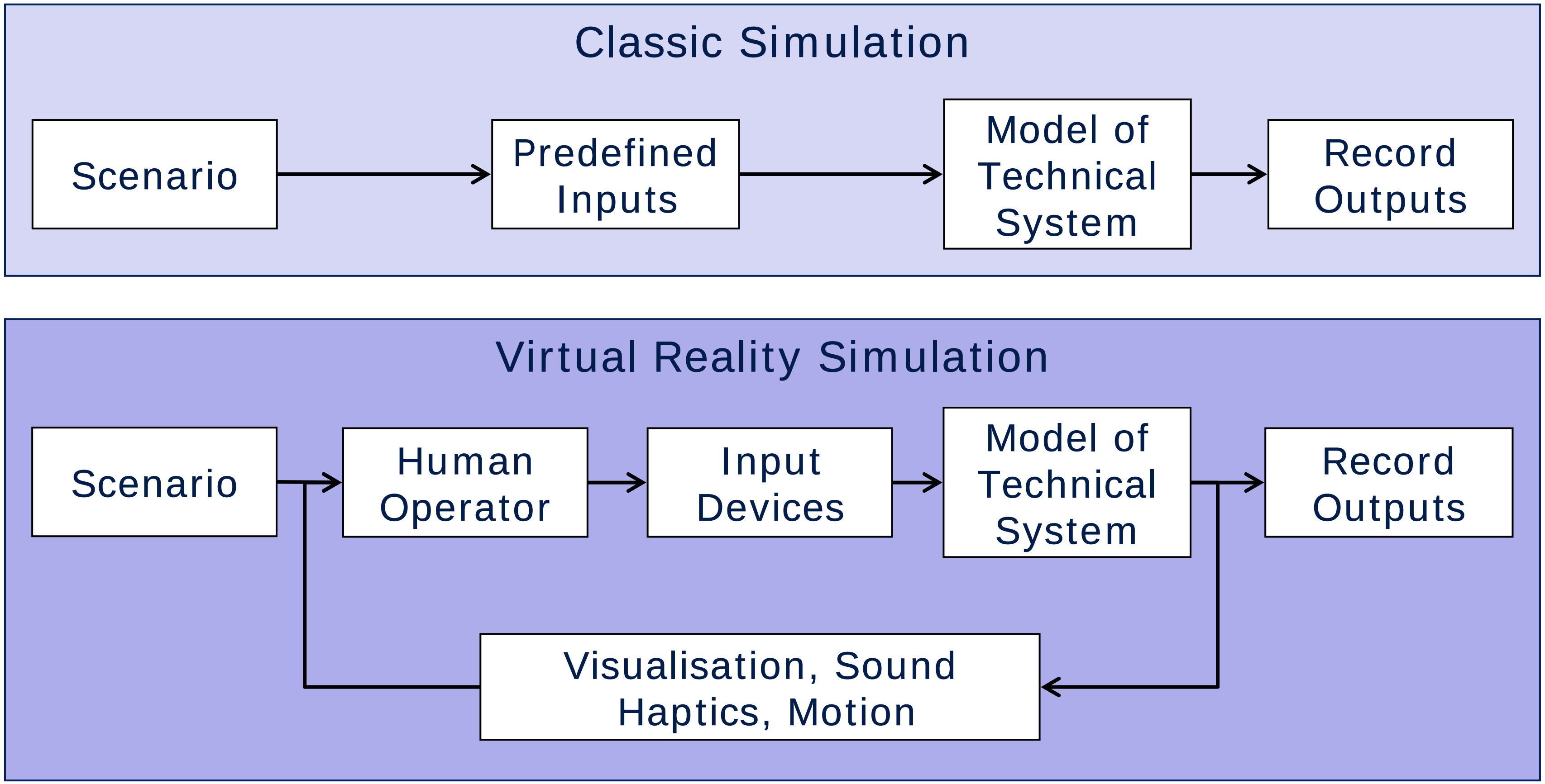
Process Simulation ● Energy Efficiency ● Human Machine Interface

TECHNISCHE UNIVERSITÄT DRESDEN

1. Introduction

1. Introduction

Why do we need Virtual Reality (VR) simulation



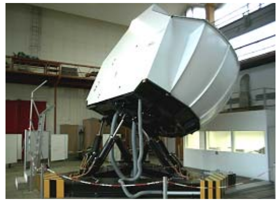
TECHNISCHE UNIVERSITÄT DRESDEN

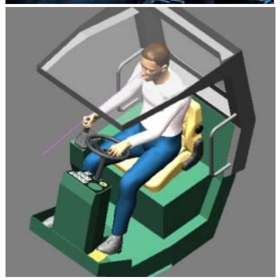
1. Introduction

Requirements on VR Simulation Software

- Realtime simulation
- Support for various input and output devices
- Realistic graphics and sound
- Easy hardware integration
- Distributed computing
- Flexibility

➡ Development of SARTURIS for interactive VR simulations

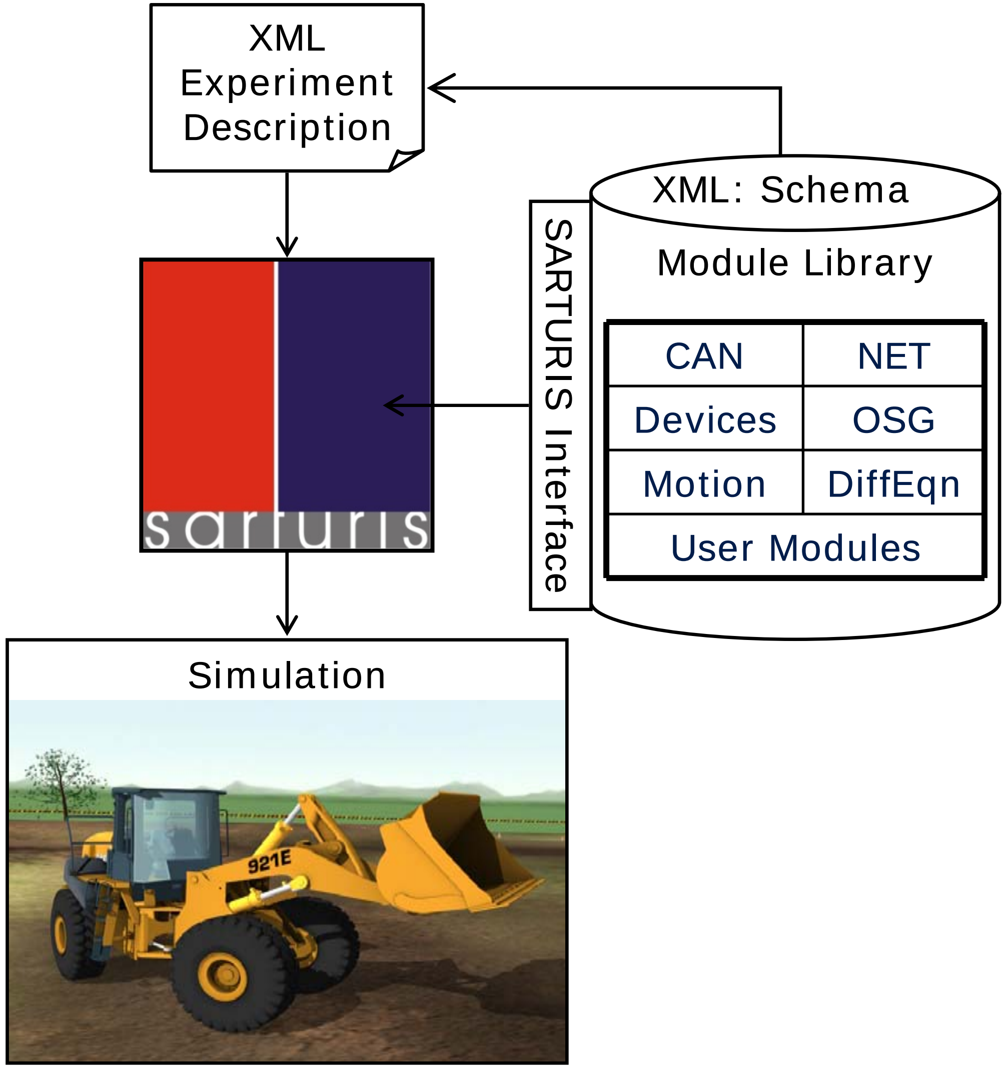




2. SARTURIS & Modelica

What is SARTURIS

- Simulation Framework
- Designed in C++
- Platform neutral
- Organised in modules
- Modules can be combined to an experiment: Experiment Description
- SARTURIS performs Simulation



2. SARTURIS & Modelica

Motivation and Idea

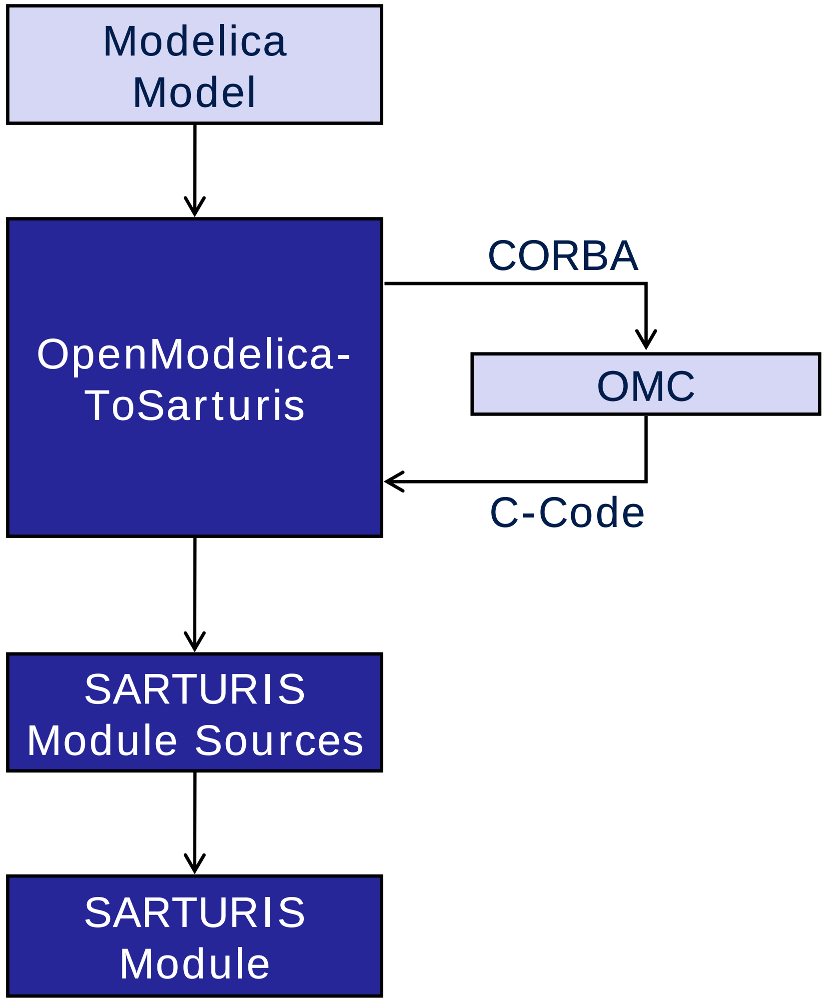
- Initial situation
  - Manual implementation of equations in C++ code
- Modelica is a great modelling language
  - Object-oriented
  - Component-based
  - Equation-based
  - Acausal
- Combine the strengths of Modelica and SARTURIS

2. SARTURIS & Modelica

TECHNISCHE UNIVERSITÄT DRESDEN

Solution

- OpenModelicaToSarturis
- Invokes OMC
- Communication via CORBA
- Parses C-code and turns it into source code for a SARTURIS module
- Build management based on CMake



3. PyMbs

Motivation

- No multibody library available for OpenModelica
- Same models are used in different simulation software
- Assembling complex systems can be quite difficult

3. PyMbs

Features

- Written in Python
- Supports different output formats including Modelica
- Includes basic visualisation (can be used to check assembly)
- Description using minimal/generalised coordinates approach

$$M\ddot{q} + h = f + \left(\frac{\partial \Phi}{\partial q}\right)^T \lambda$$

$$\Phi(q) = 0$$

- Implicit or explicit handling of kinematic loops
- Symbolic simplification of equations of motion

TECHNISCHE UNIVERSITÄT DRESDEN

3. PyMbs

Internal Structure

Input

- Python Source Code
- Defining
  - Bodies
  - Joints
  - Load Elements
  - Sensors

Equations

- Obtaining Equations of Motion from holonomic system
- Either using a recursive or an explicit scheme
- Taking care of kinematic loops

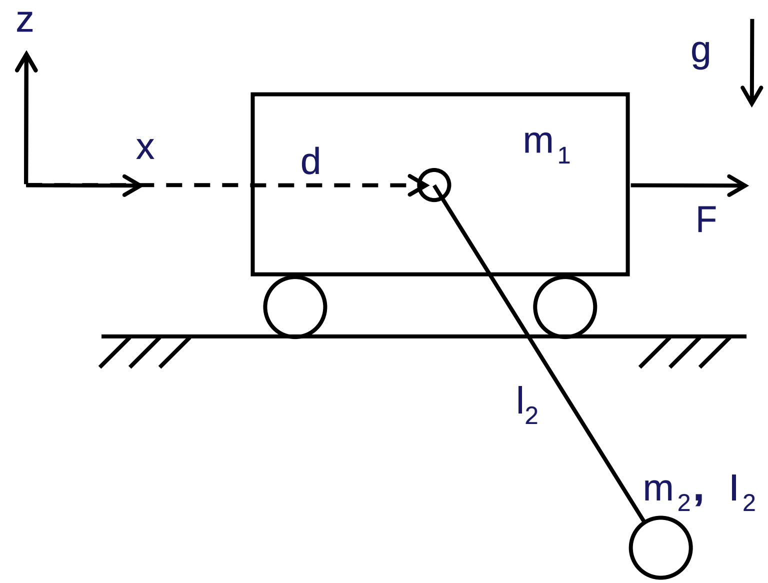
Analysis

- Determining causal order of equations
- Simplifying Expressions
  - $a+0=a$
  - $a \cdot 0=0$
  - $a \cdot 1=a$

Output

- Writing simulation code:
  - Modelica
  - MATLAB
  - Python
- Visualisation
  - Python
  - MATLAB

Example: Crane Crab



1
```
# Set up a new MbsSystem
world = MbsSystem( [0,0,-1] )

# Define Input and Parameters
F  = world.addInput( 'Force'     , 'F' )
m1 = world.addParam( 'mass 1'    , 'm1', 10 )
m2 = world.addParam( 'mass 2'    , 'm2', 1 )
l2 = world.addParam( 'length'    , 'l2', 1 )
I2 = world.addParam( 'inertia 2', 'I2', 1/12 )
```

2
```
# Define Bodies and Coordinate Systems
crab = world.addBody( 'Crab', mass=m1 )
load = world.addBody( 'Load', mass=m2, inertia=diag([0,I2,0]), cg=[0,0,-l2] )
```

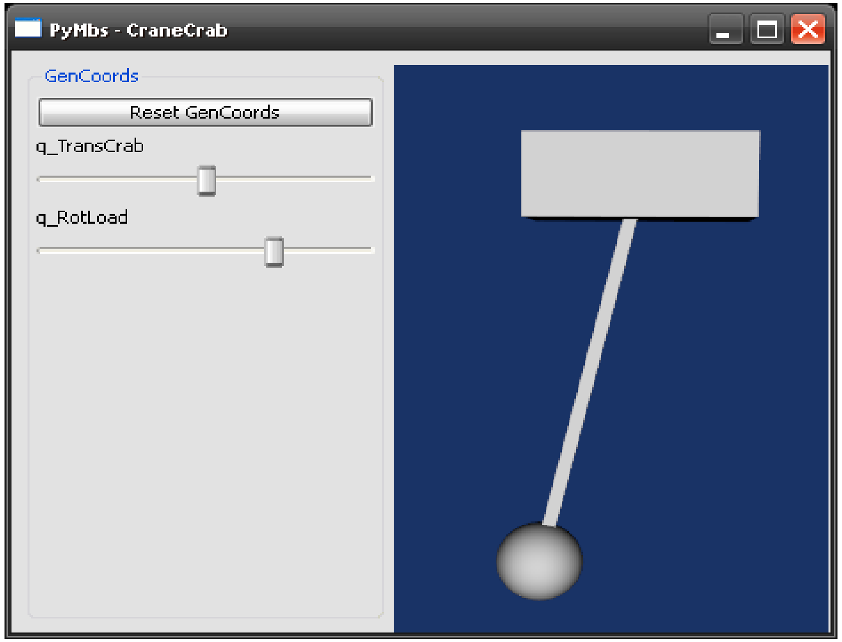
3
```
# Connect Bodies Through Joints
world.addJoint( 'TransCrab', world, crab, 'Tx', startVals=1 )
world.addJoint( 'RotLoad'   , crab , load.joint, 'Ry' )
```

4
```
# Add Sensors and Force Elements
world.addLoad( 'DrivingForce', 'PtPForce', crab, world, F )
world.addSensor( 'Position', 'Distance',   crab, world, 'd' )
```

5
```
# Calculate Equations of Motion and Generate Code
world.genEquations( explicit = True )
world.genCode( 'mo', 'CraneCrab_PyMbs' )
```

Example: Crane Crab

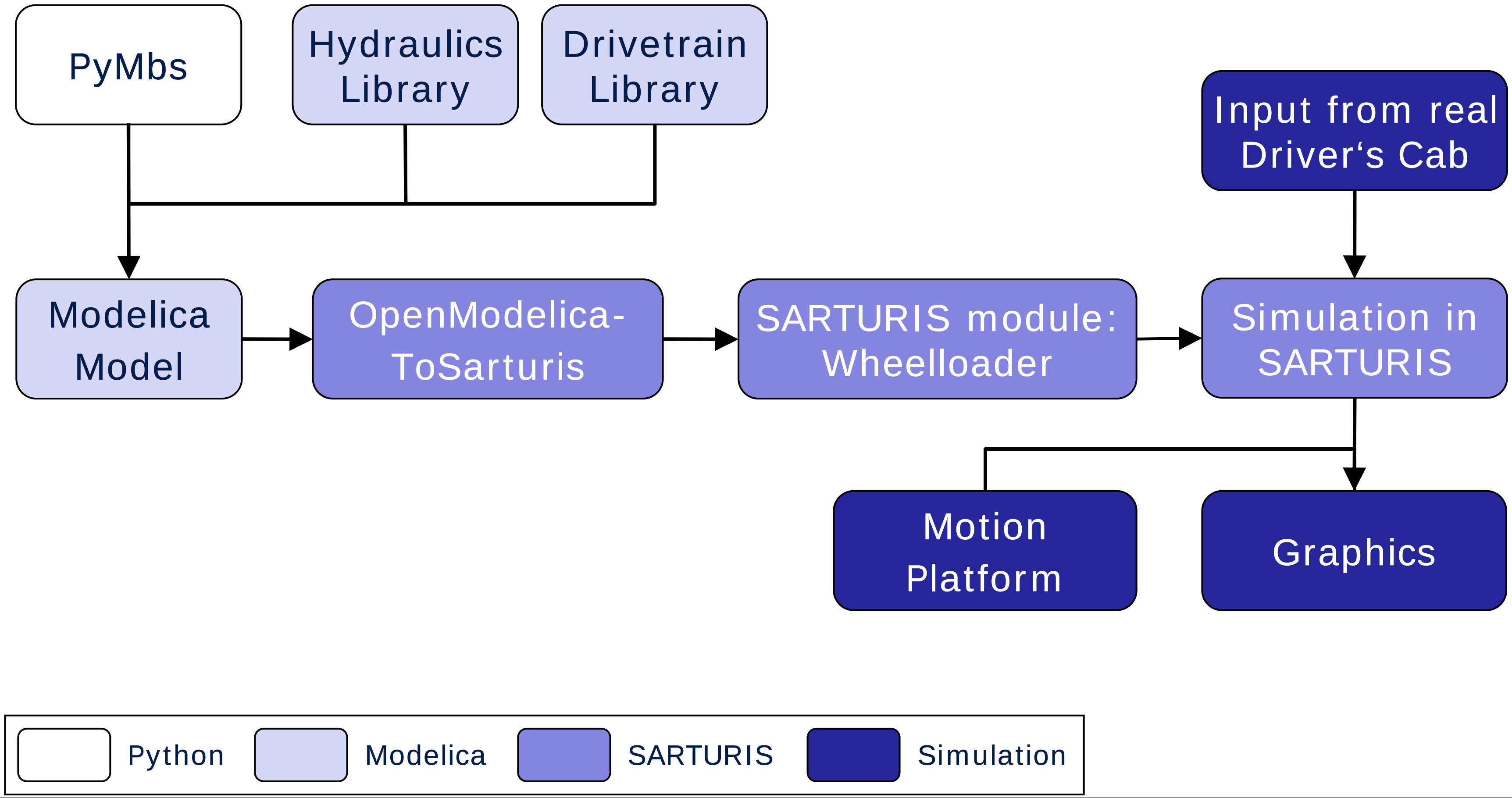
```
// This file was generated by PyMbs
partial model CraneCrab_PyMbs
  Real[2] q (start={1,0})
  Real[2] qd (start={0,0})
  ...
equation
  M = { { m1+m2, l2*m2*sin(q[2]) },
        { l2*m2*sin(q[2]), I2+m2*l2^2 }};
  h = { l2*m2*qd[2]^2*cos(q[2]), 0 };
  f_gravity = { 0, -g*l2*m2*cos(q[2]) };
  WF_DrivingForce = { q[1]/abs(q[1]), 0 };
  f_ext = F*WF_DrivingForce;
  f = f_ext+f_gravity;
  der(q) = qd;
  M*der(qd) + h = f;
  d = { abs(q[1]), qd[1]*q[1]/abs(q[1]) };
end CraneCrab_PyMbs;
```




```
model CraneCrab extends CraneCrab_PyMbs;
  import Modelica.Mechanics.Translational.*;
  Interfaces.Flange_b flange;
equation
  flange.s = d[1];
  flange.f = F;
end CraneCrab;
```

4. Demonstration

Model of a Wheel Loader



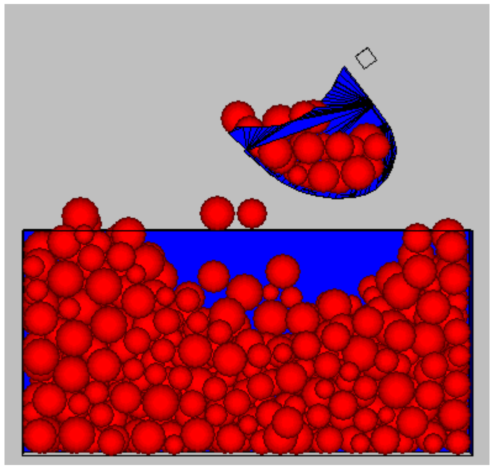
4. Demonstration

Wheel Loader in Action

5. Conclusion

Future Work

- Process simulation (concrete, soil, ...)
- Extending realtime capabilities
- Improving calculation
  - Parallel code
  - Distributed computing



5. Conclusion

Results

- SARTURIS has been designed for interactive realtime simulation
- Integration of Modelica improves modelling
- PyMbs allows comfortable description of multibody systems supporting different output formats

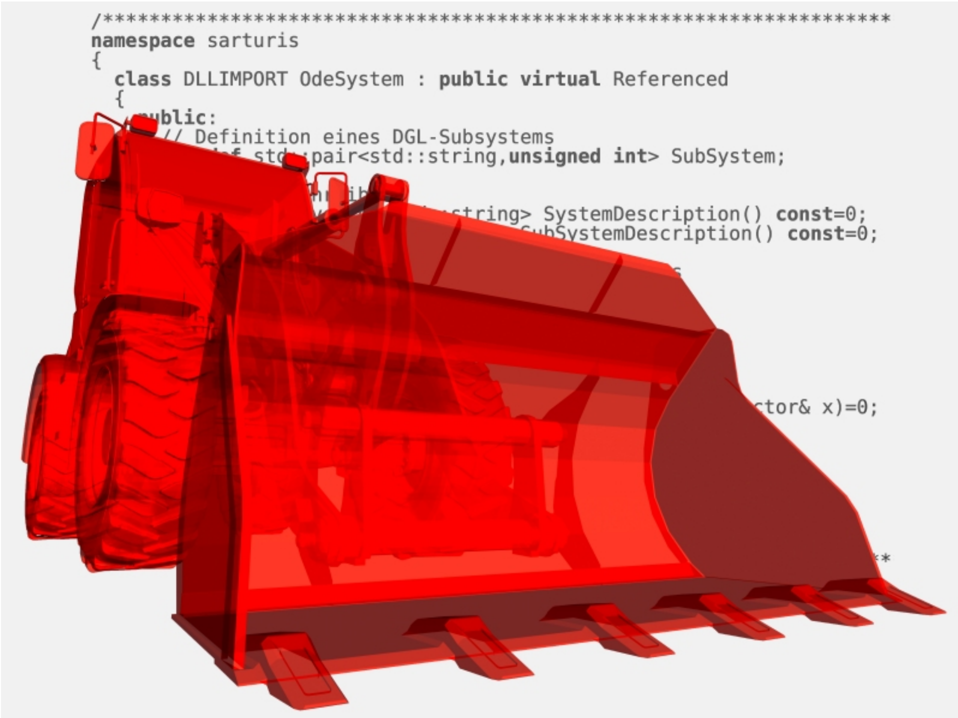
TECHNISCHE UNIVERSITÄT DRESDEN

The End

»Wissen schafft Brücken.«

2. SARTURIS & Modelica

History of SARTURIS



- Motion platform in 2003 (monolithic system)
- Started out as a research project* in 2004
- First release in 2007
- Constant improvements
- Central component in current research

Federal Ministry of Education and Research

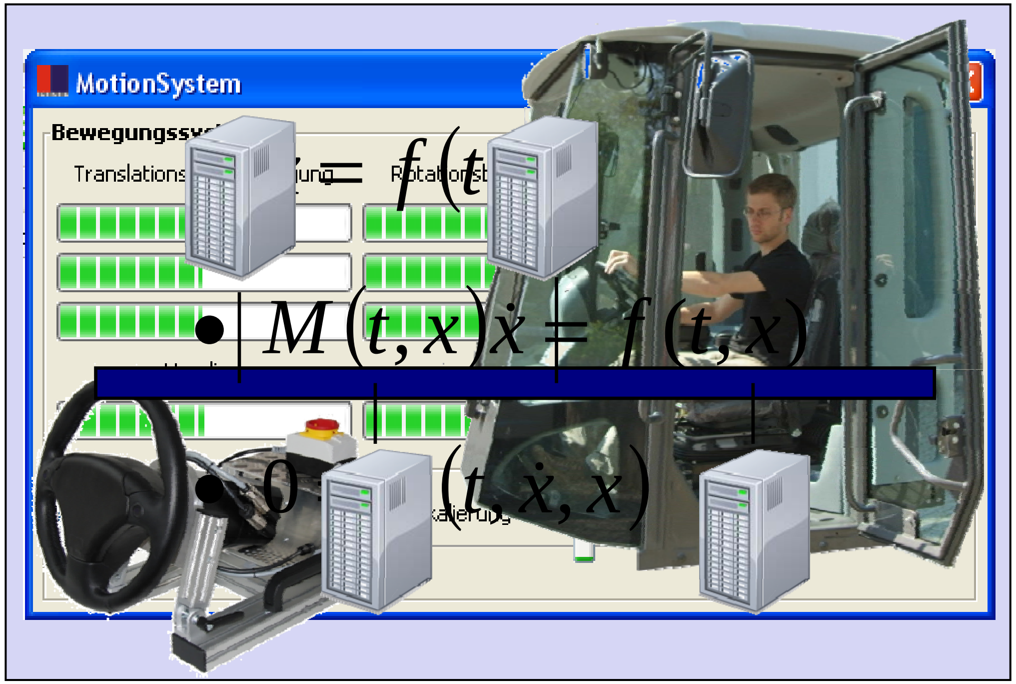
* funded by BMBF

TECHNISCHE UNIVERSITÄT DRESDEN

2. SARTURIS & Modelica

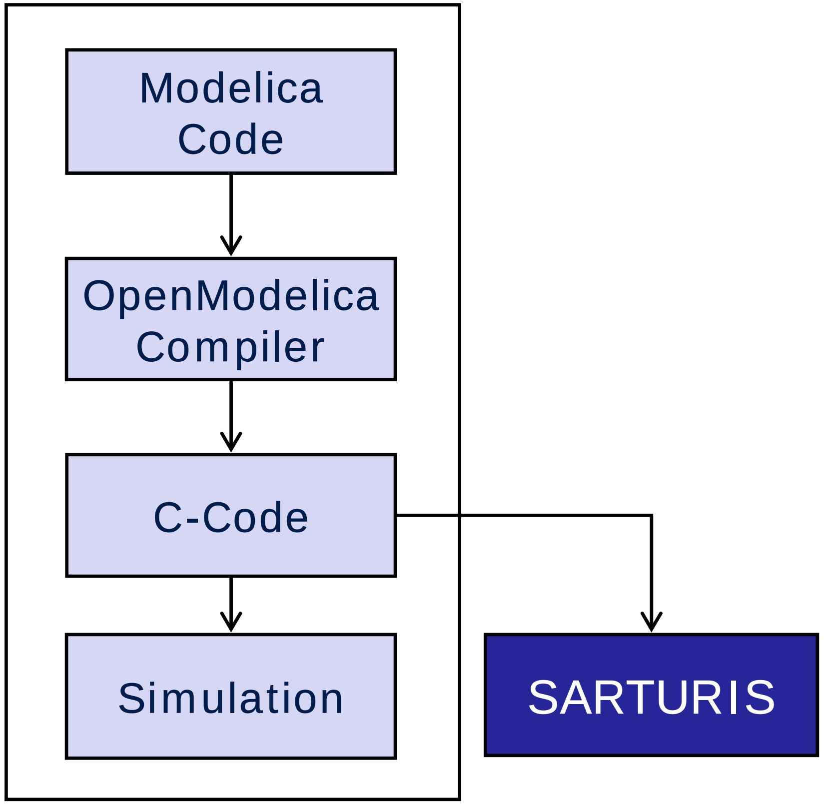
Selection of Available Modules

- OpenSceneGraph
- GUI
- Devices
- Network
- Differential Equations



2. SARTURIS & Modelica

Method of Integration



- Usage of existing Modelica software
- OpenModelica was chosen
  - Open source
  - Option to take part in development
  - Easy to integrate
  - Can be used for teaching

4. Demonstration

Wheel Loader

- CNH 921 E
- Assessment of novel control systems
- 40 states
  - Multibody
  - Hydraulics
  - Drivetrain
- Real Driver's Cab
- Motion Platform

Outside View

View from Inside the Cabin