Modelica Association Project
“Distributed Co-Simulation Protocol”

Martin Krammer
DCP MAP Leader
martin.krammer@v2c2.at

ACOSAR
ADVANCED CO-SIMULATION OPEN SYSTEM ARCHITECTURE

ITEA 3

FFG
Austrian
Research Promotion Agency

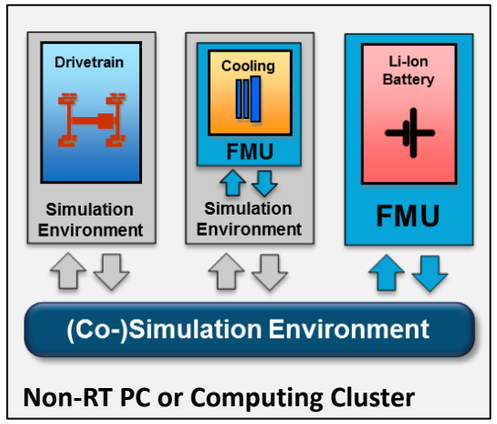
Federal Ministry
of Education
and Research

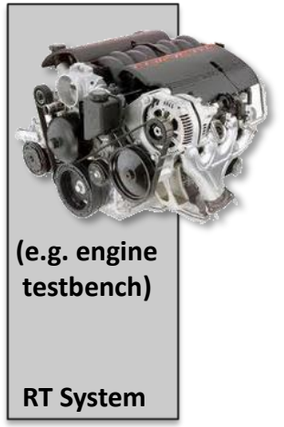
Outline

- Introduction
- The Distributed Co-Simulation Protocol (DCP)
  - Communication Protocol
  - Architecture Description
  - Operating Modes
  - State Machine
  - Exchange of Input and Output Data
  - Use Case
- The Future of DCP

Motivation

- The Functional Mock-up Interface (FMI, MODELISAR project) standardizes integration of simulation models, tools and solvers
- But what about distributed setups?





- Until now, this is done manually

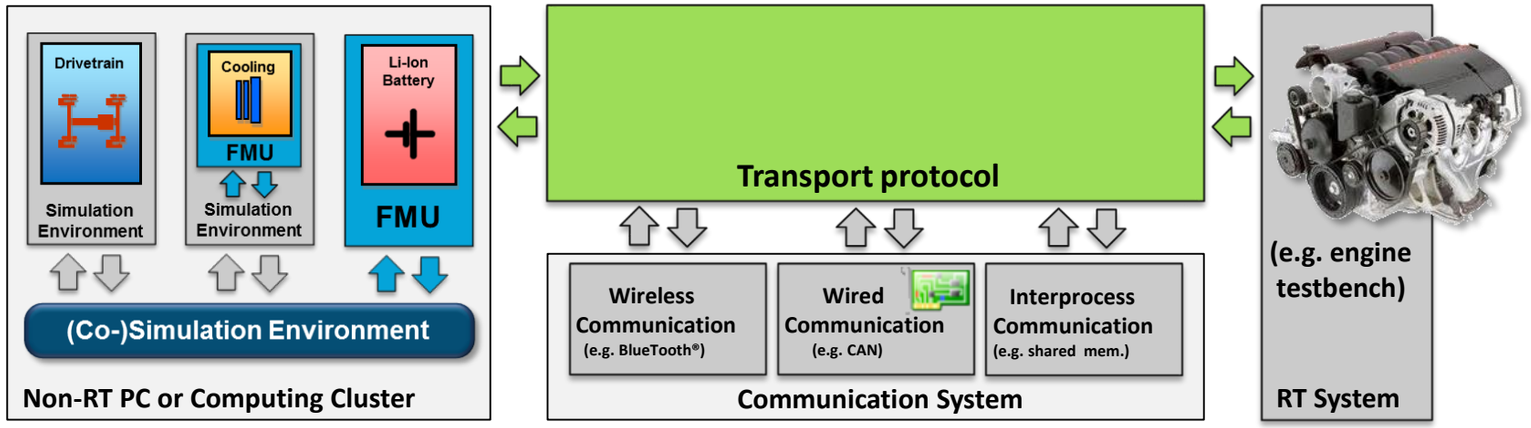
Motivation

- The Functional Mock-up Interface (FMI, MODELISAR project) standardizes integration of simulation models, tools and solvers
- But what about distributed setups?

- Until now, this is done manually

Motivation

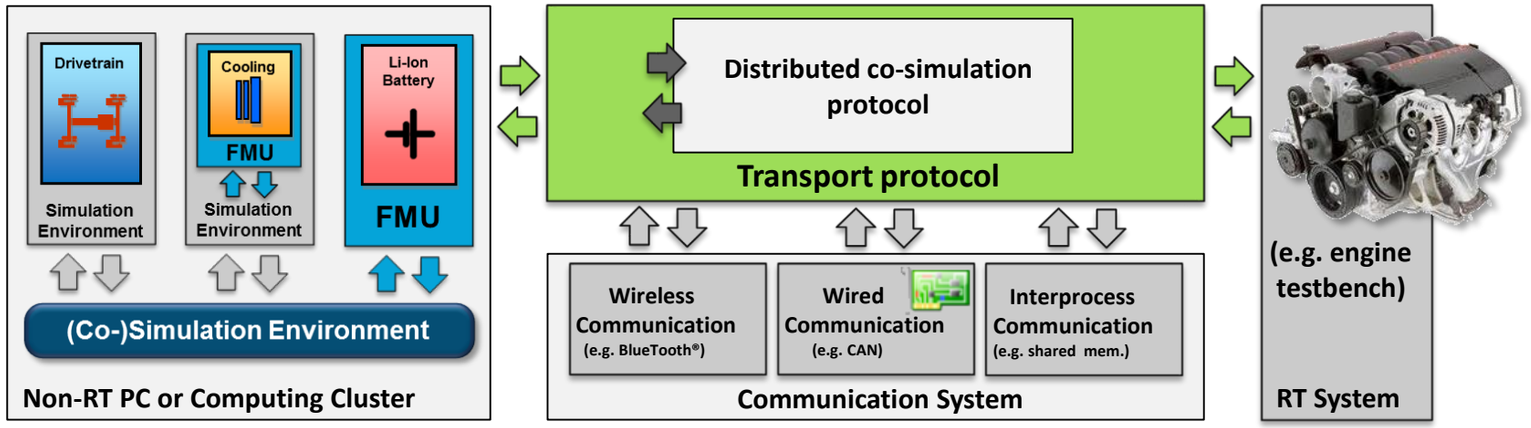
- The Functional Mock-up Interface (FMI, MODELISAR project) standardizes integration of simulation models, tools and solvers
- But what about distributed setups?



- Until now, this is done manually

Motivation

- The Functional Mock-up Interface (FMI, MODELISAR project) standardizes integration of simulation models, tools and solvers
- But what about distributed setups?

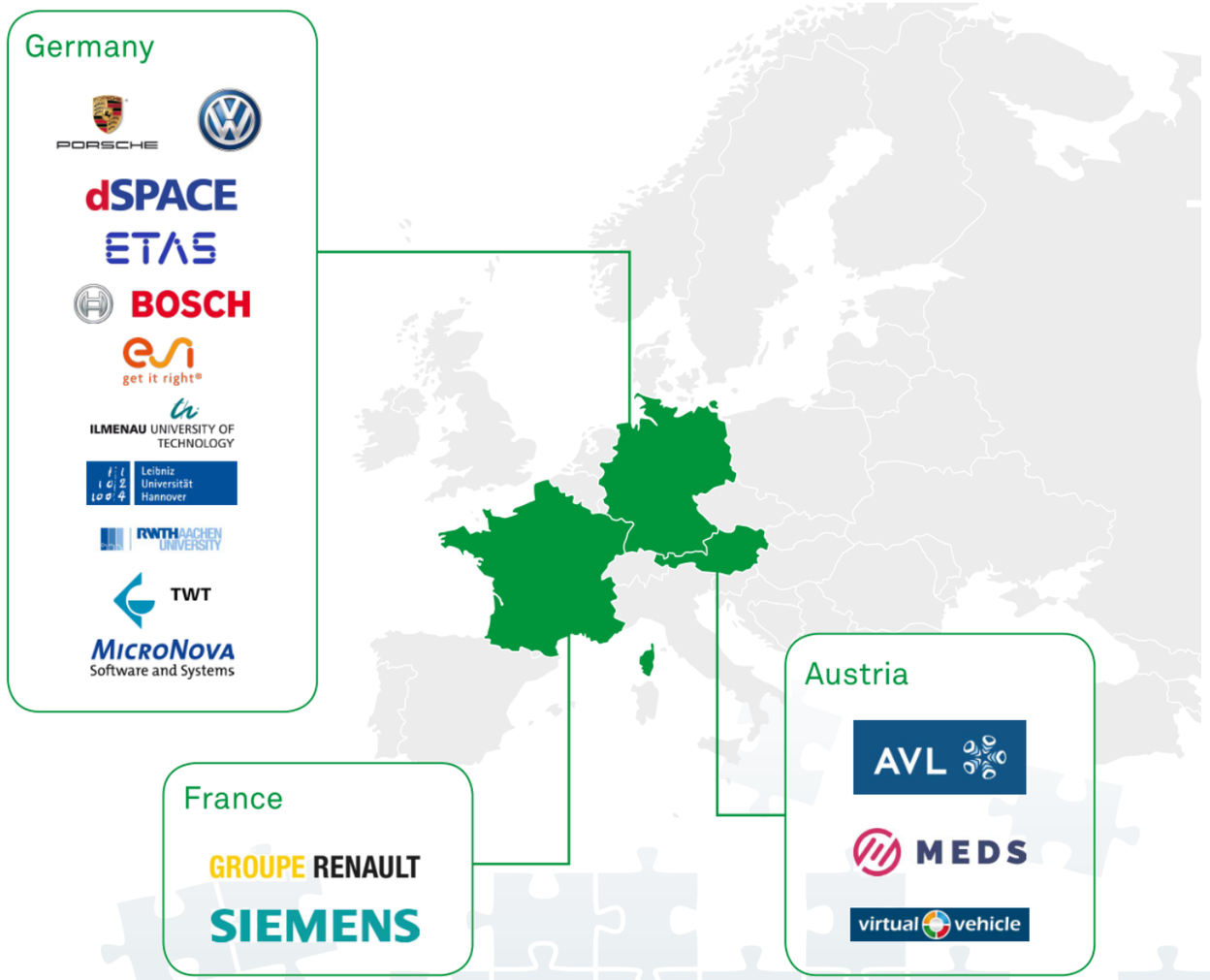


- Until now, this is done manually

History - The ACOSAR Project

ACOSAR
ADVANCED CO-SIMULATION OPEN SYSTEM ARCHITECTURE

- The ACOSAR project
  - *Advanced Co-Simulation Open System Architecture*
  - Duration: 09/2015-08/2018
  - Costs: 8,123k€
  - Effort: 60 PY
- ACOSAR focuses on integration of
  - Real-time and real-time, and
  - Real-time and non-real-time systems
- **Primary goal: Negotiate technical specification of communication protocol intended for standardization**

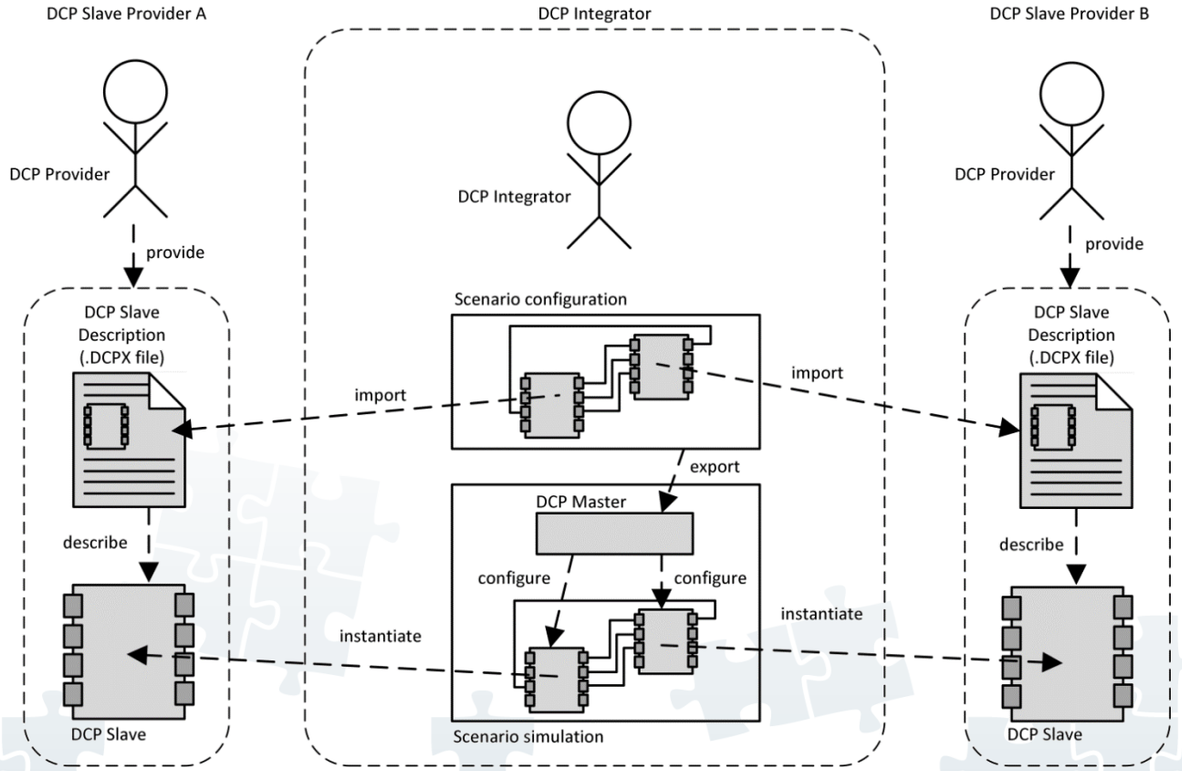


The Distributed Co-Simulation Protocol

- Main design aspects
  - Interoperability
    - Define a communication protocol
    - Goal: Pursue standardization with a recognized standardization body
  - Compatibility
    - Support a broad range of systems, from small microcontrollers to large test rigs
    - Targets: Low overhead, low memory footprint
  - Integration
    - Develop methodology for application in development processes
    - Master-Slave concept
  - Communication
    - Support multiple transport protocols
    - Initially: UDP, CAN, USB, Bluetooth, and EtherCAT
  - Economy
    - Reduce development time
    - Decrease computing cost
    - Accelerate time-to-market

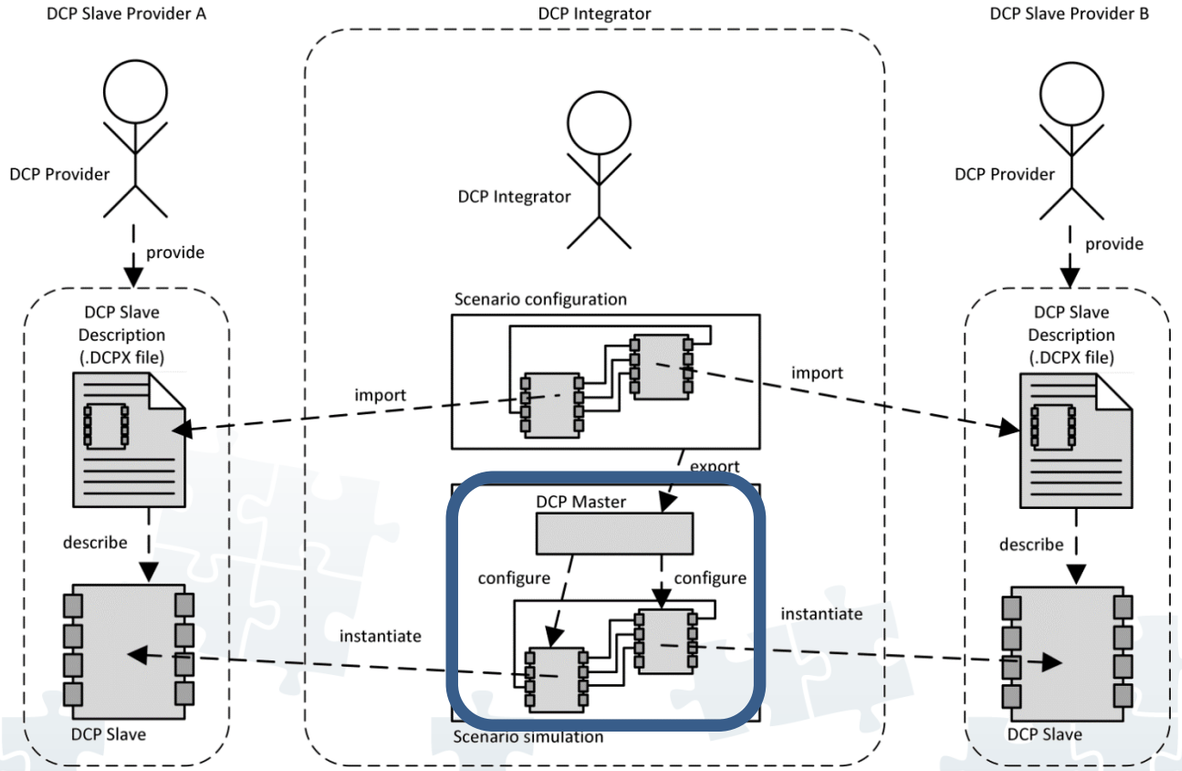
The Distributed Co-Simulation Protocol

- Default integration methodology
- Relies on DCP slave description file (.dcpx)
- Defines provider-integrator relationship



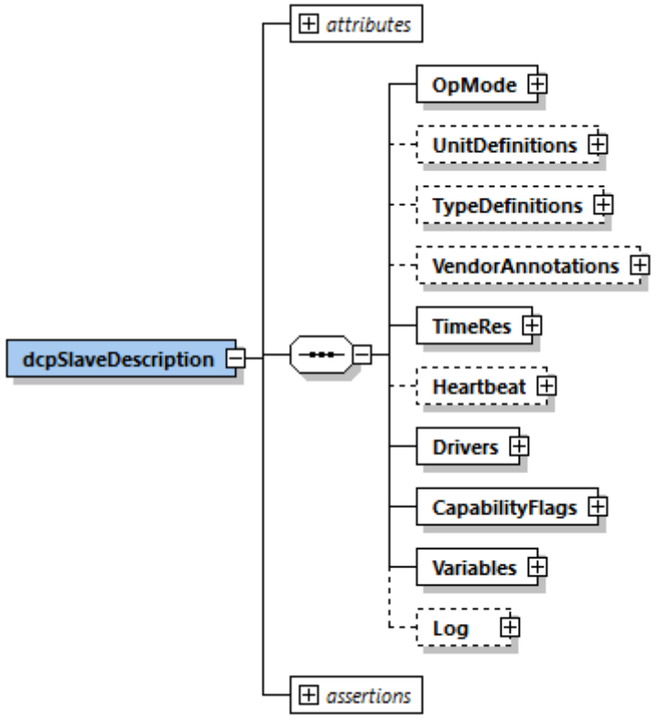
The Distributed Co-Simulation Protocol

- Default integration methodology
- Relies on DCP slave description file (.dcpx)
- Defines provider-integrator relationship



The Distributed Co-Simulation Protocol

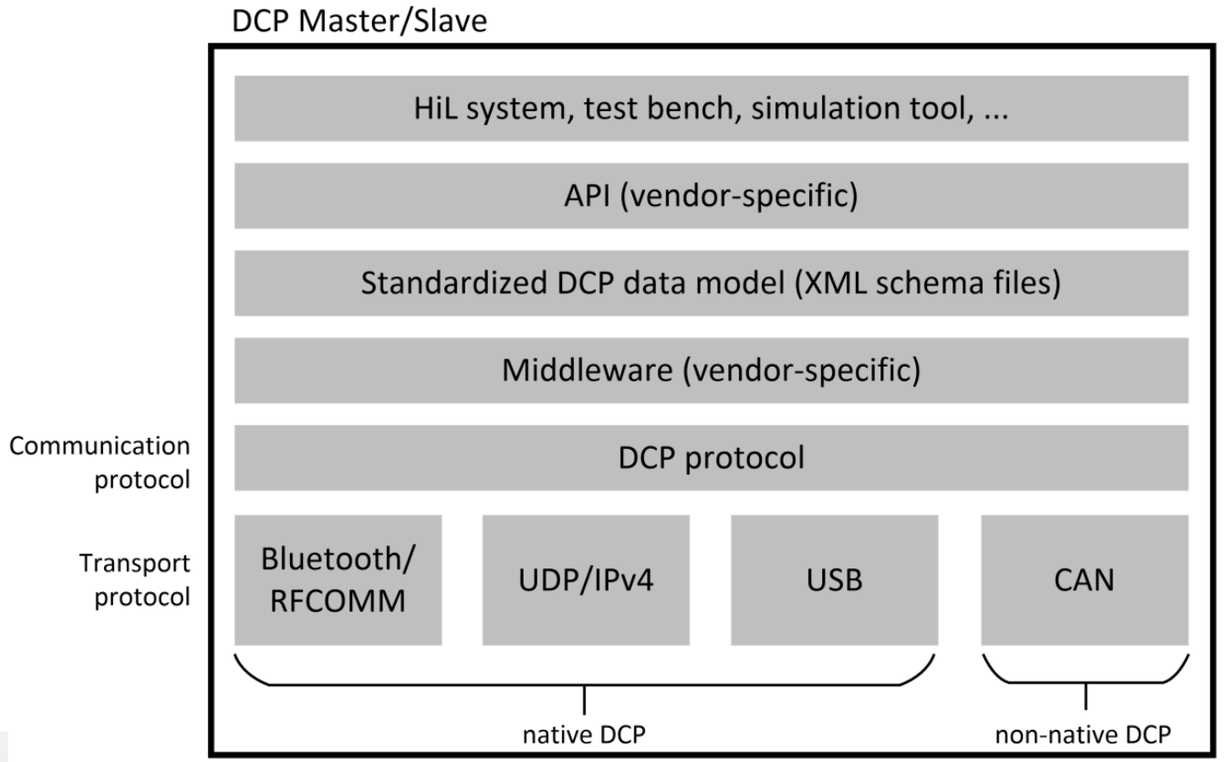
- DCP Slave Description
  - DCP slave description accompanies DCP slave
  - Specified as XSD 1.1 schema definition
  - XML instance .dcpx File
- Assertions
  - Used to enforce specifications
  - Avoid incorrect definitions
- Available transformation strips assertions and generates XSD 1.0 schema



Operating mode
Units
Types
Vendor specific annotations
Time resolution
Heartbeat definitions
Transport protocol
Capability flags
Variables (inputs/outputs/parameters)
Logging definitions

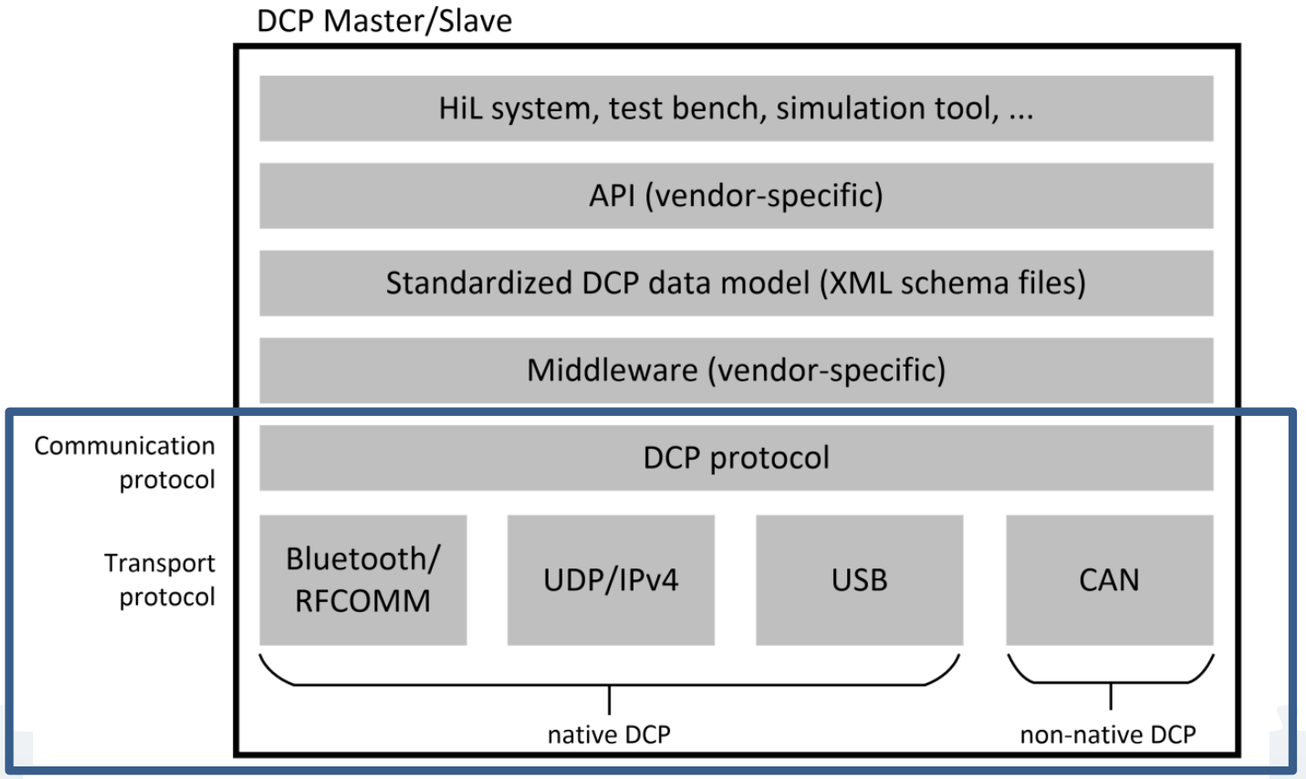
The Distributed Co-Simulation Protocol

- Architecture Description



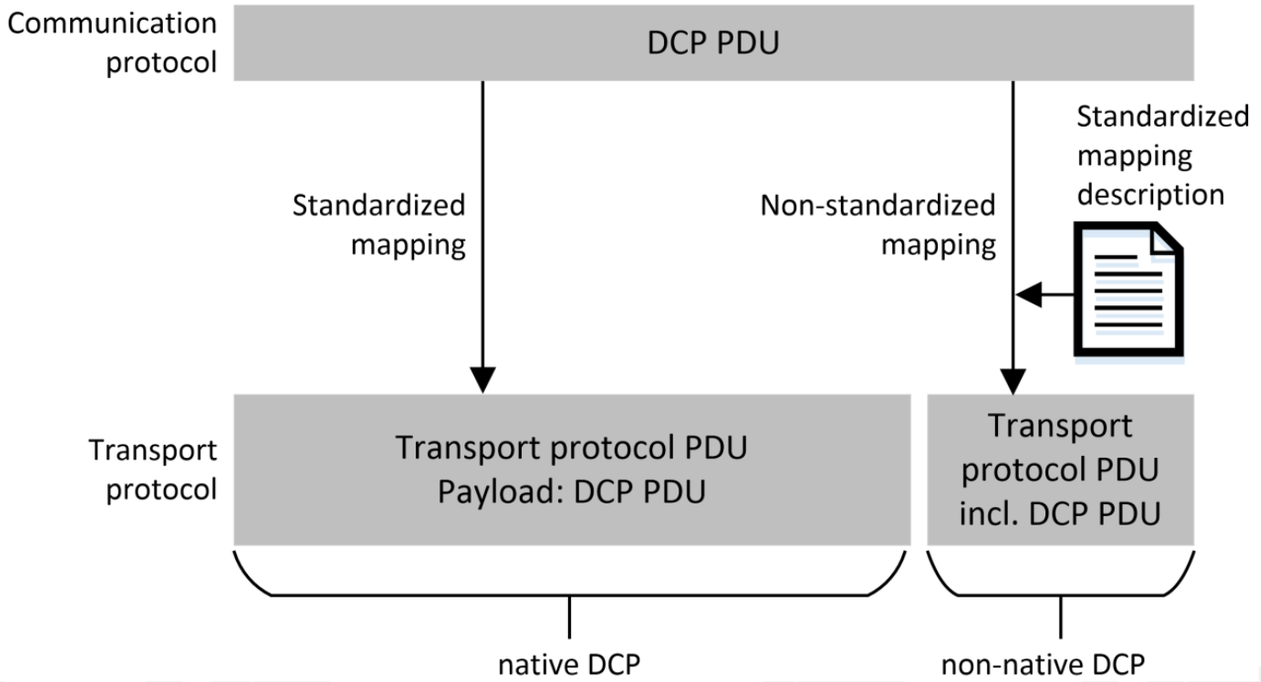
The Distributed Co-Simulation Protocol

- Architecture Description



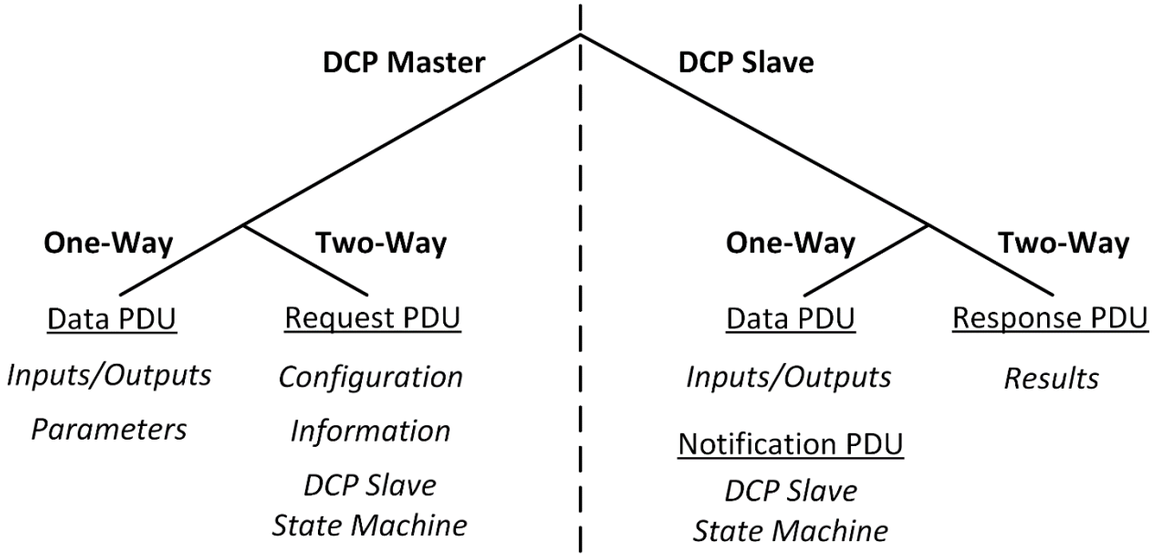
The Distributed Co-Simulation Protocol

- Architecture Description



The Distributed Co-Simulation Protocol

- Taxonomy of Protocol Data Units (PDU)
  - „PDU Families“



								DCP Frame															
								type_id	pdu_seq_id	resp_seq_id	sender	receiver	param_id	data_id	pos	target_vr	source_vr	source_data_type	transport_protocol	state_id	numerator	denominator	steps	op_mode
Protocol Data Unit (PDU)	Control	Request	Configuration (CFG)	CFG_set_time_res				0x20	y			y									y	y		
				CFG_set_steps				0x21	y			y		y									y	
				CFG_config_input				0x22	y			y		y	y	y		y						
				CFG_config_output				0x23	y			y		y	y		y							
				CFG_config_clear				0x24	y			y												
				CFG_set_target_network_information				0x25	y			y		y					y					
				CFG_set_source_network_information				0x26	y			y		y					y					
				CFG_set_parameter				0x27	y			y						y						
				CFG_set_config_tunable_parameter				0x28	y			y	y		y			y						
				CFG_set_param_network_information				0x29	y			y	y						y					
				CFG_set_logging				0x2A	y			y												
				CFG_set_scope				0x2B	y			y		y										
			State change (STC)	STC_register				0x01	y			y								y				y
				STC_unregister				0x02	y			y								y				
				STC_configure				0x03	y			y								y				
				STC_initialize				0x04	y			y								y				
				STC_run				0x05	y			y								y				
				STC_reinitialize				0x06	y			y								y				
				STC_do_step				0x07	y			y								y			y	
				STC_send_outputs				0x08	y			y								y				
				STC_stop				0x09	y			y								y				
				STC_reset				0x0A	y			y								y				
			Information (INF)	INF_state				0x80	y			y												
				INF_error				0x81	y			y												
				INF_log				0x82	y			y												
			Response (RSP)	RSP_ack				0xB0		y	y													
				RSP_nack				0xB1		y	y													
				RSP_state_ack				0xB2		y	y									y				
				RSP_error_ack				0xB3		y	y													
				RSP_log_ack				0xB4		y	y													
		Notification (NTF)		NTF_state_changed				0xE0			y									y				
				NTF_log				0xE1			y													
		Data (DAT)		DAT_input_output				0xF0	y					y										
				DAT_parameter				0xF1	y				y											

The Distributed Co-Simulation Protocol

- Operating Modes
  - The DCP covers three different time domains

Operating mode	Description
Soft real-time (SRT)	Synchronous to absolute time, tolerant to RT violations
Hard real-time (HRT)	Synchronous to absolute time, intolerant to RT violations
Non-real-time (NRT)	Independent of absolute time

The Distributed Co-Simulation Protocol

- DCP slave state machine for simulation control
- A typical simulation cycle
  1. Registration
  2. Configuration
  3. Initialization
  4. Run/Compute
  5. Stop
  6. (Error)

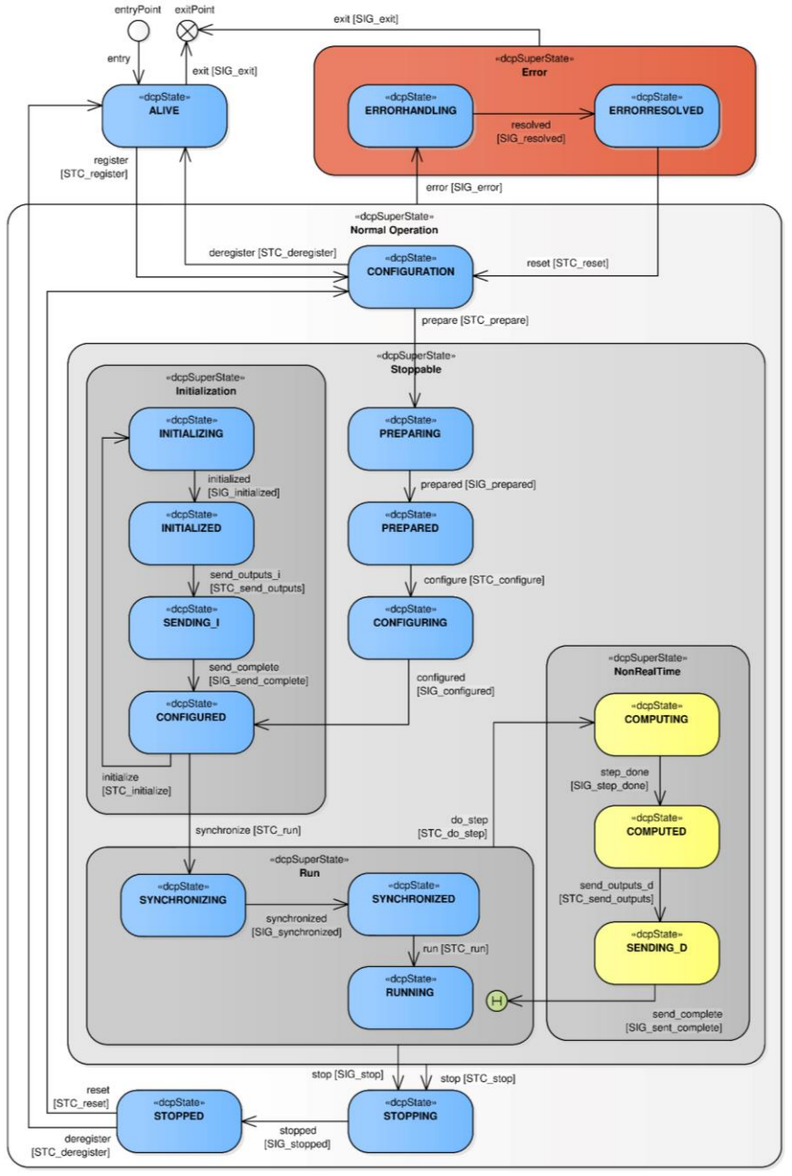


Image source: DCP specification v1.0 RC 2

The Distributed Co-Simulation Protocol

- DCP slave state machine for simulation control
- A typical simulation cycle
  1. Registration
  2. Configuration
  3. Initialization
  4. Run/Compute
  5. Stop
  6. (Error)

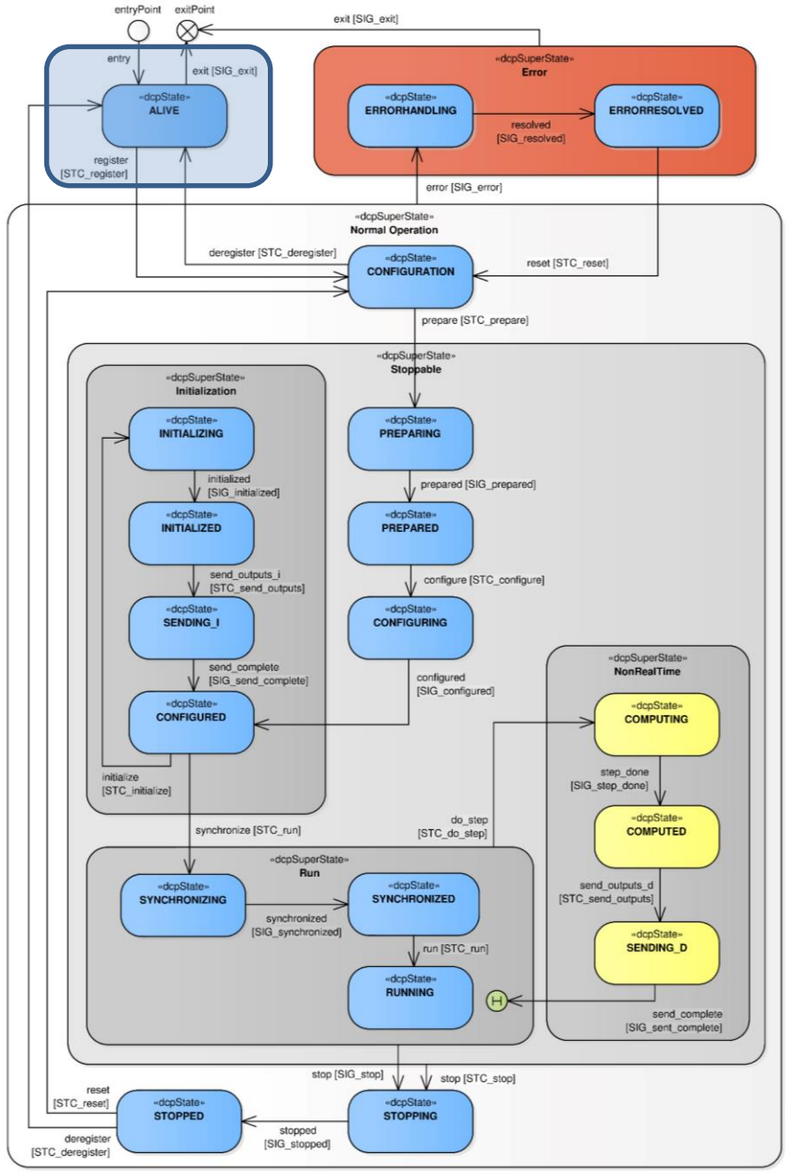


Image source: DCP specification v1.0 RC 2

The Distributed Co-Simulation Protocol

- DCP slave state machine for simulation control
- A typical simulation cycle
  1. Registration
  2. Configuration
  3. Initialization
  4. Run/Compute
  5. Stop
  6. (Error)

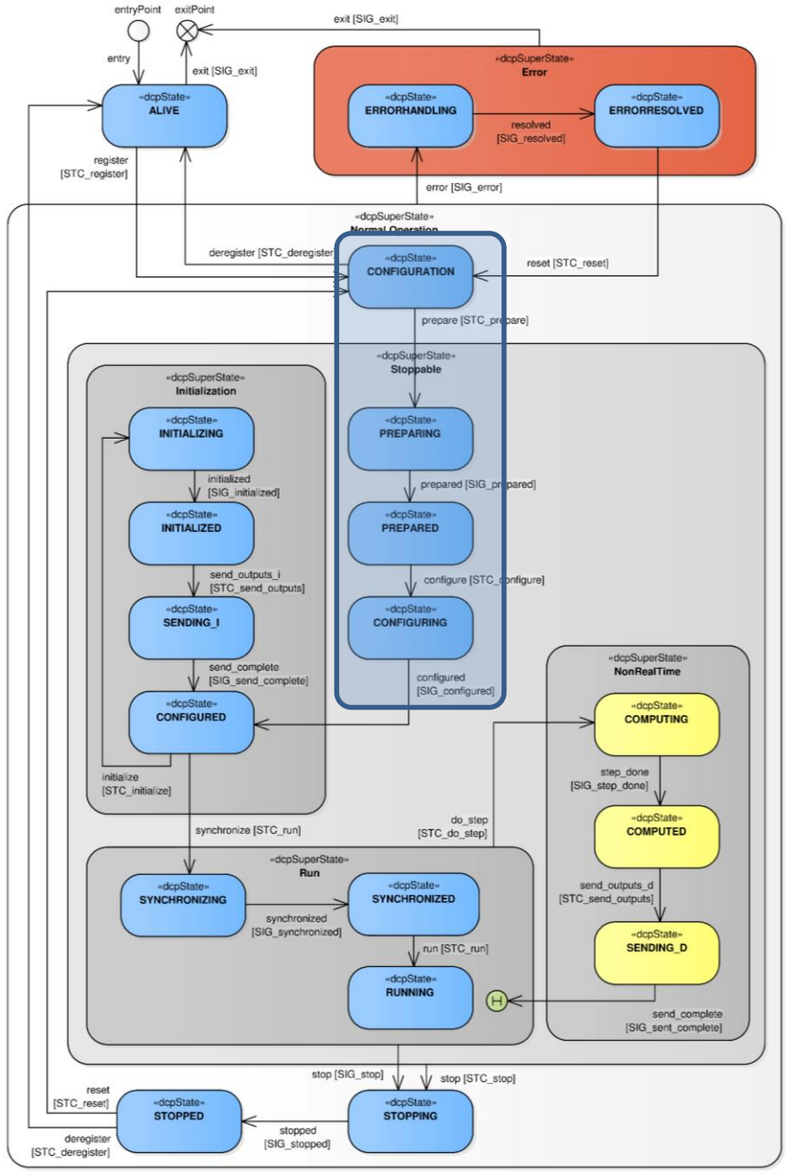


Image source: DCP specification v1.0 RC 2

The Distributed Co-Simulation Protocol

- DCP slave state machine for simulation control
- A typical simulation cycle
  1. Registration
  2. Configuration
  3. Initialization
  4. Run/Compute
  5. Stop
  6. (Error)

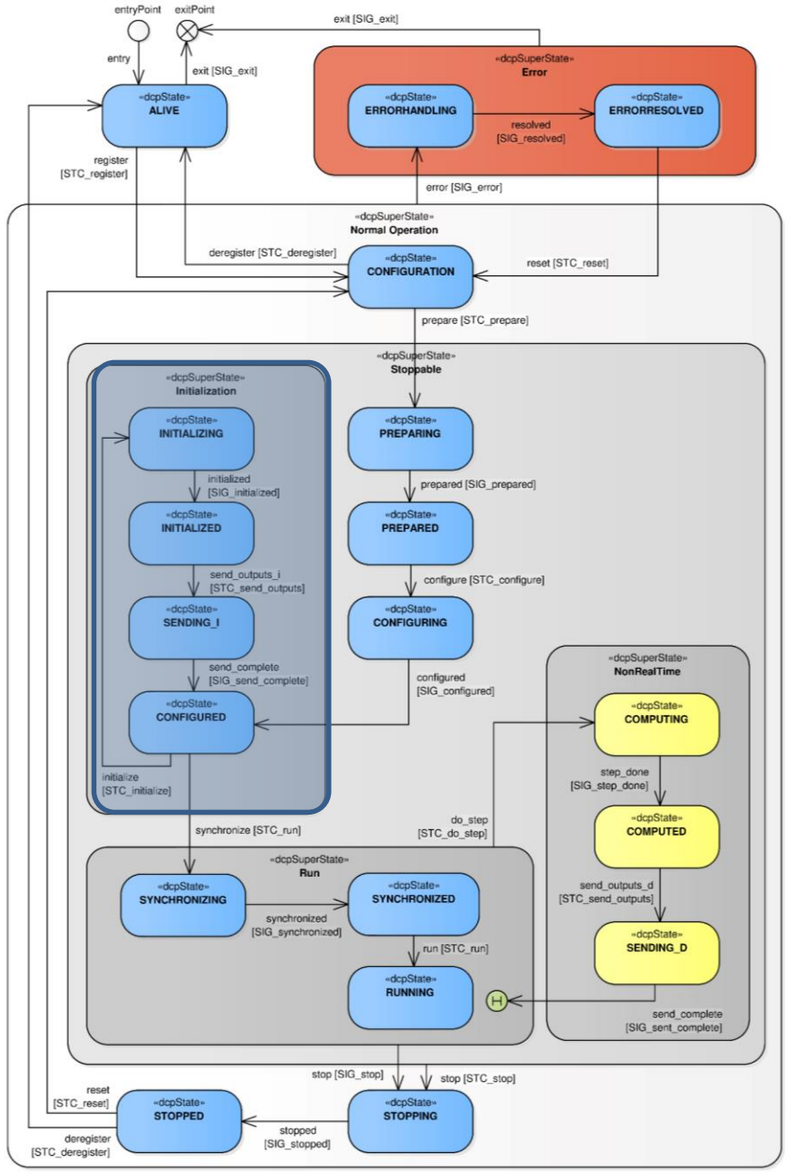


Image source: DCP specification v1.0 RC 2

The Distributed Co-Simulation Protocol

- DCP slave state machine for simulation control
- A typical simulation cycle
  1. Registration
  2. Configuration
  3. Initialization
  4. Run/Compute
  5. Stop
  6. (Error)

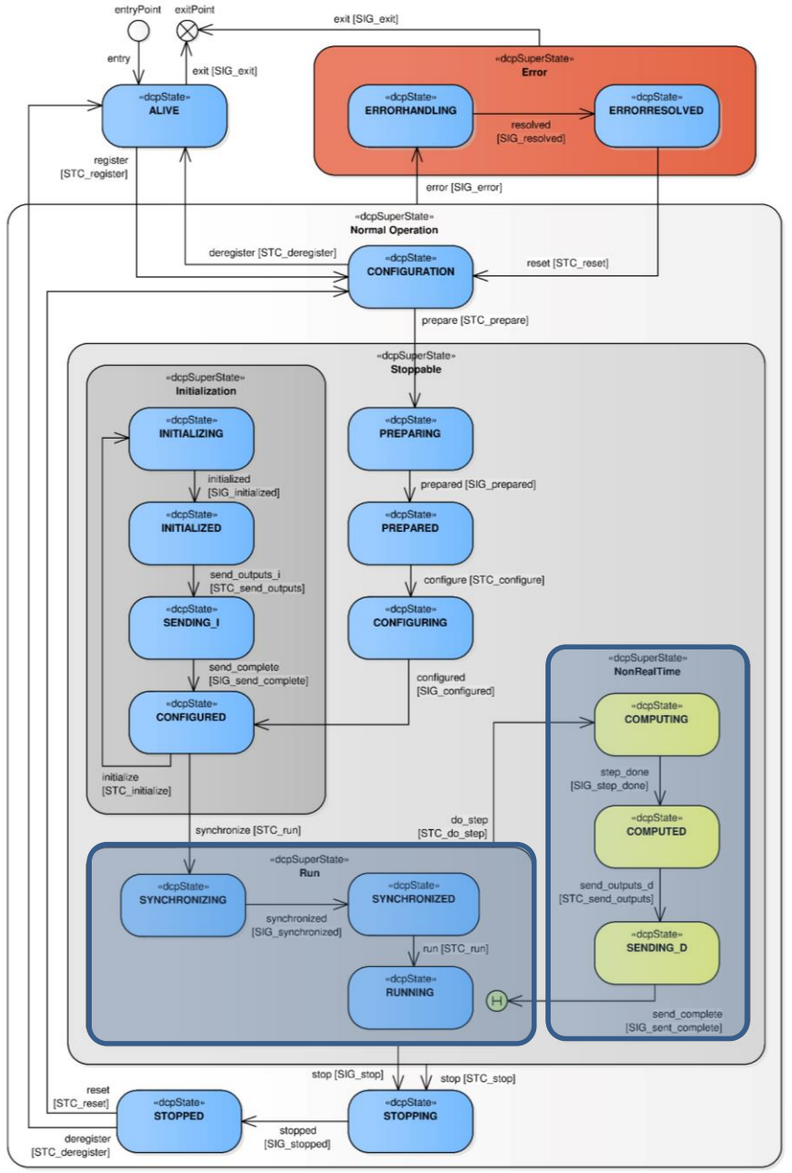


Image source: DCP specification v1.0 RC 2

The Distributed Co-Simulation Protocol

- DCP slave state machine for simulation control
- A typical simulation cycle
  1. Registration
  2. Configuration
  3. Initialization
  4. Run/Compute
  5. Stop
  6. (Error)

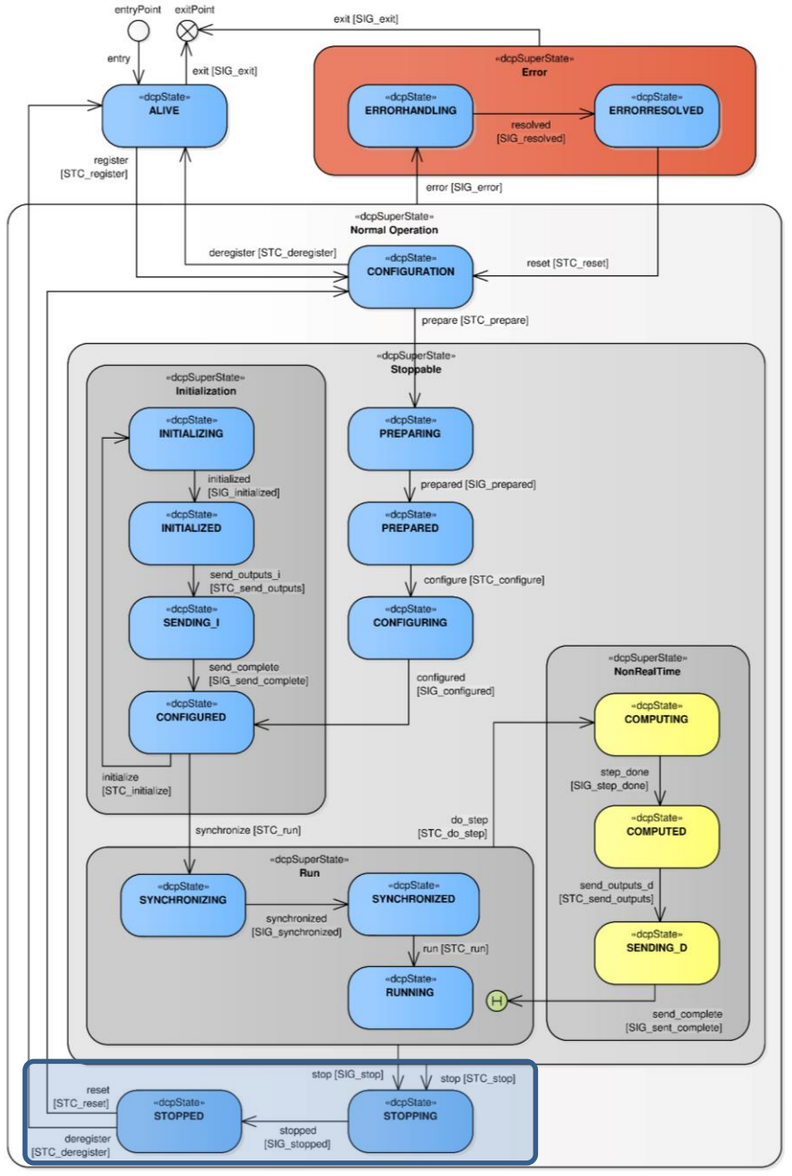


Image source: DCP specification v1.0 RC 2

The Distributed Co-Simulation Protocol

- DCP slave state machine for simulation control
- A typical simulation cycle
  1. Registration
  2. Configuration
  3. Initialization
  4. Run/Compute
  5. Stop
  6. (Error)

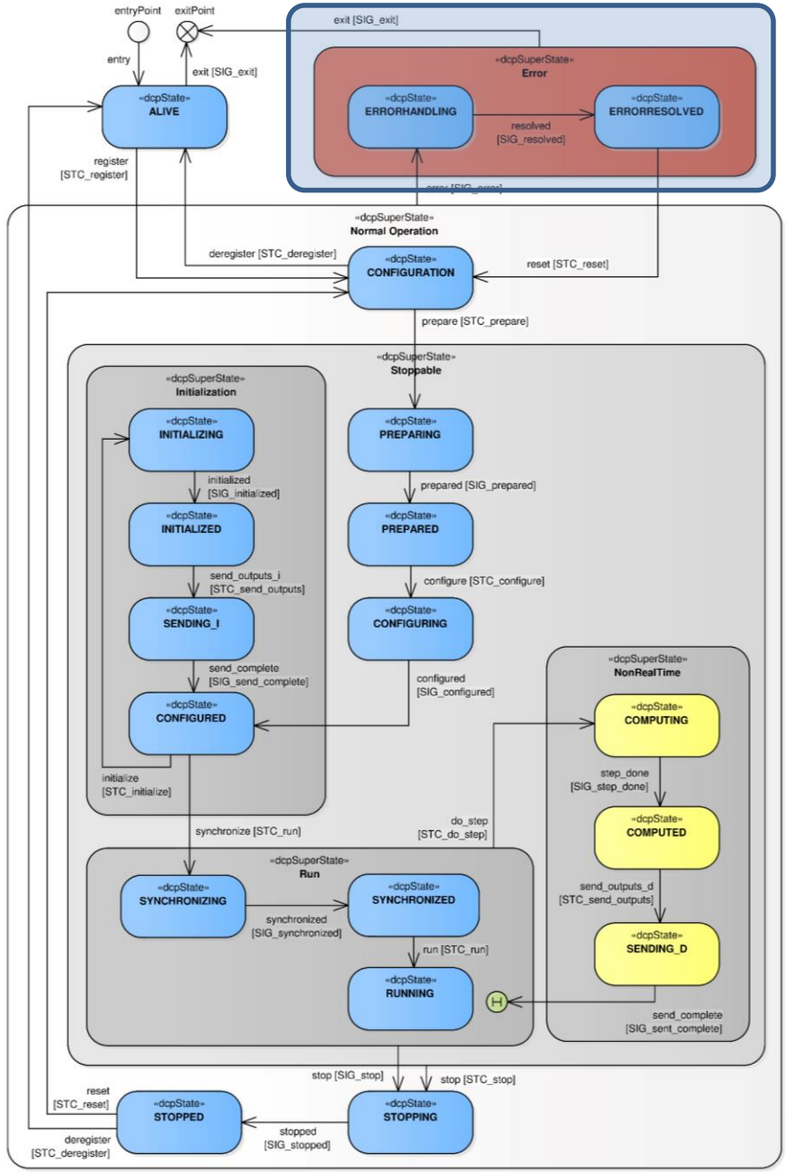


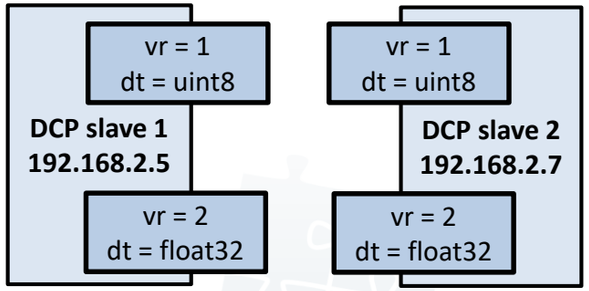
Image source: DCP specification v1.0 RC 2

The Distributed Co-Simulation Protocol

- Exchange of data during simulation phase
  1. Configuration must be generated by DCP master
  2. Configuration must be rolled out to DCP slaves prior to simulation
- *Zero run time overhead* during simulation
- Example:

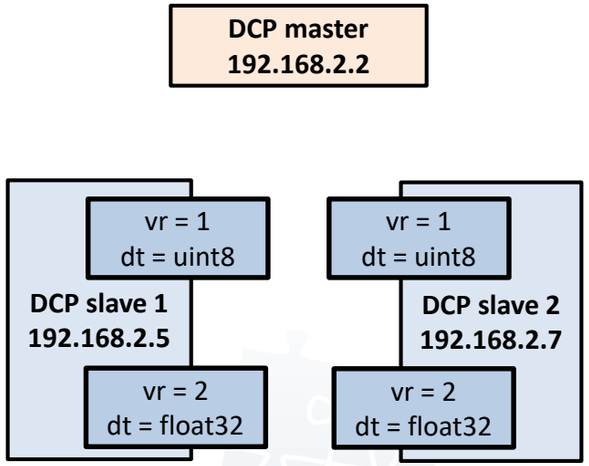
The Distributed Co-Simulation Protocol

- Exchange of data during simulation phase
  1. Configuration must be generated by DCP master
  2. Configuration must be rolled out to DCP slaves prior to simulation
- *Zero run time overhead* during simulation
- Example:



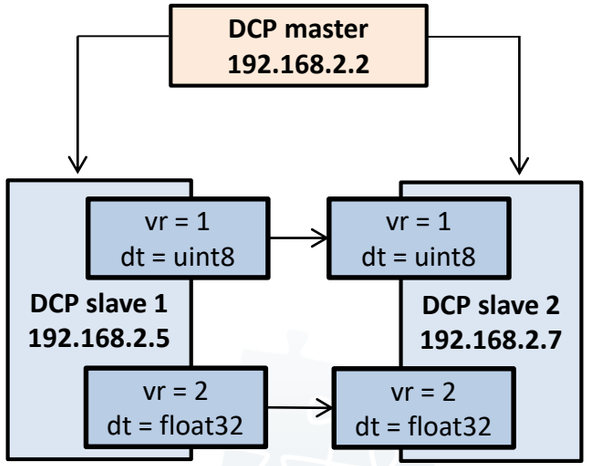
The Distributed Co-Simulation Protocol

- Exchange of data during simulation phase
  1. Configuration must be generated by DCP master
  2. Configuration must be rolled out to DCP slaves prior to simulation
- *Zero run time overhead* during simulation
- Example:



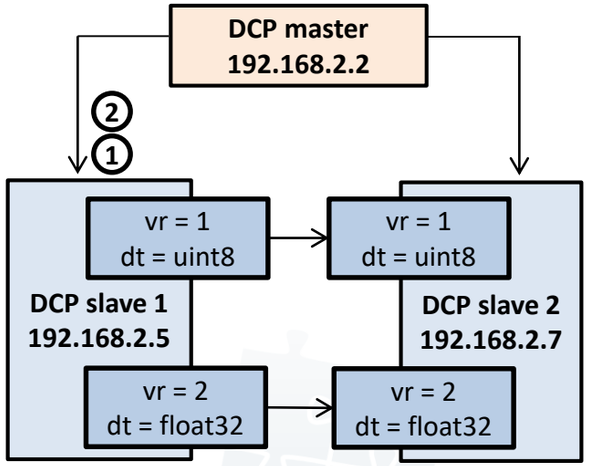
The Distributed Co-Simulation Protocol

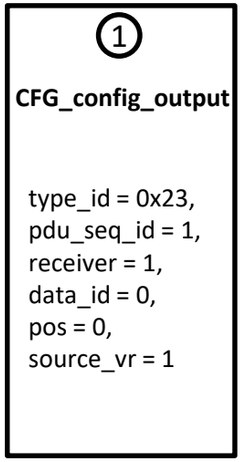
- Exchange of data during simulation phase
  1. Configuration must be generated by DCP master
  2. Configuration must be rolled out to DCP slaves prior to simulation
- *Zero run time overhead* during simulation
- Example:

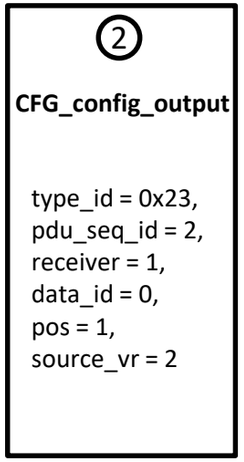


The Distributed Co-Simulation Protocol

- Exchange of data during simulation phase
  1. Configuration must be generated by DCP master
  2. Configuration must be rolled out to DCP slaves prior to simulation
- *Zero run time overhead* during simulation
- Example:

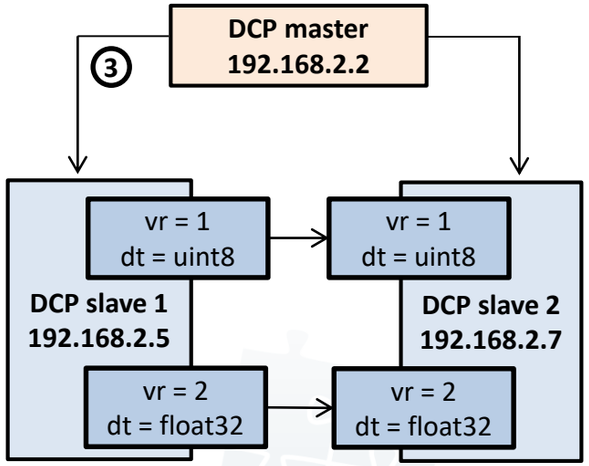






The Distributed Co-Simulation Protocol

- Exchange of data during simulation phase
  1. Configuration must be generated by DCP master
  2. Configuration must be rolled out to DCP slaves prior to simulation
- *Zero run time overhead* during simulation
- Example:



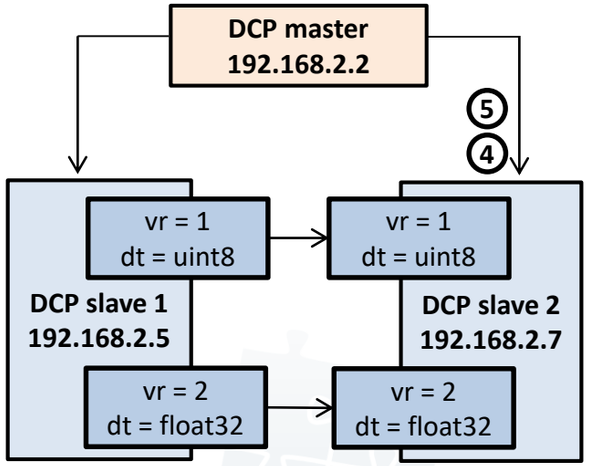
3

CFG_set_target_network_information

type_id = 0x25,
pdu_seq_id = 3,
receiver = 1,
data_id = 0,
transport_protocol = UDP,
target_ip_address = 192.168.2.7,
target_port = 2048

The Distributed Co-Simulation Protocol

- Exchange of data during simulation phase
  1. Configuration must be generated by DCP master
  2. Configuration must be rolled out to DCP slaves prior to simulation
- *Zero run time overhead* during simulation
- Example:



4

CFG_config_input

type_id = 0x22,
pdu_seq_id = 1,
receiver = 2,
data_id = 0,
pos = 0,
target_vr = 1,
source_data_type = uint8

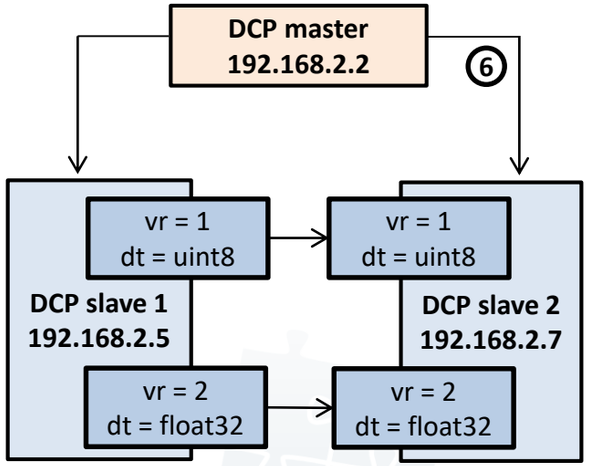
5

CFG_config_input

type_id = 0x22,
pdu_seq_id = 2,
receiver = 2,
data_id = 0,
pos = 1,
target_vr = 2,
source_data_type = float32

The Distributed Co-Simulation Protocol

- Exchange of data during simulation phase
  1. Configuration must be generated by DCP master
  2. Configuration must be rolled out to DCP slaves prior to simulation
- *Zero run time overhead* during simulation
- Example:



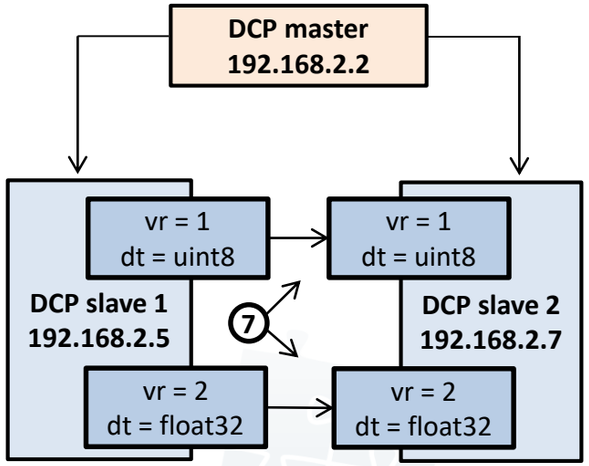
6

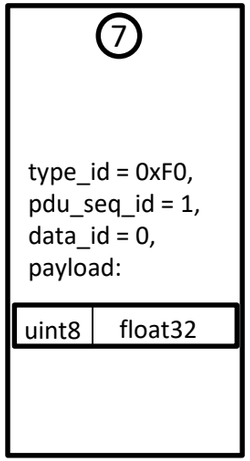
CFG_set_source_network_information

type_id = 0x26,
pdu_seq_id = 3,
receiver = 2,
data_id = 0,
transport_protocol = UDP,
source_port = 2049

The Distributed Co-Simulation Protocol

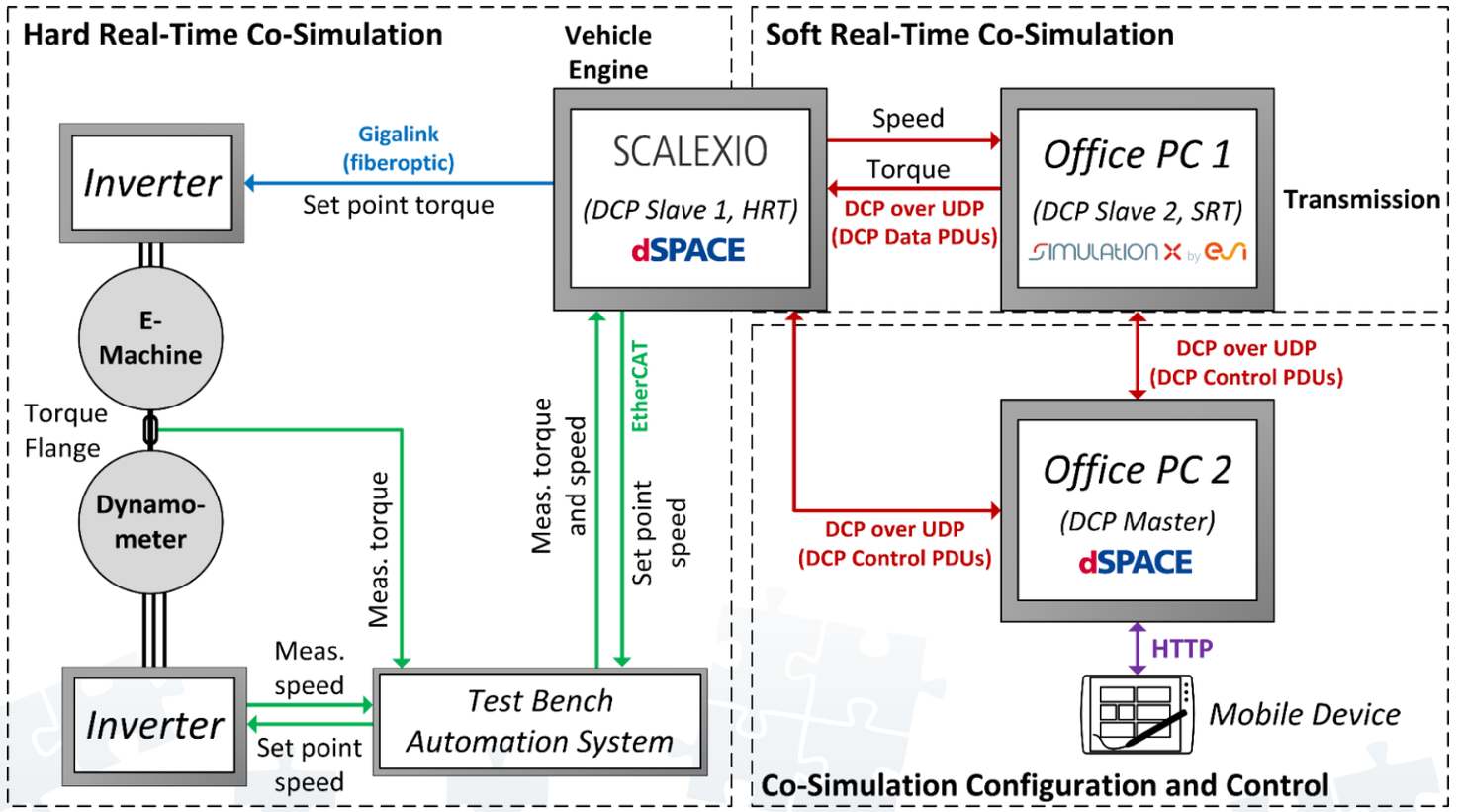
- Exchange of data during simulation phase
  1. Configuration must be generated by DCP master
  2. Configuration must be rolled out to DCP slaves prior to simulation
- *Zero run time overhead* during simulation
- Example:





The Distributed Co-Simulation Protocol

- Use case by dSPACE, RWTH Aachen, ESI-ITI

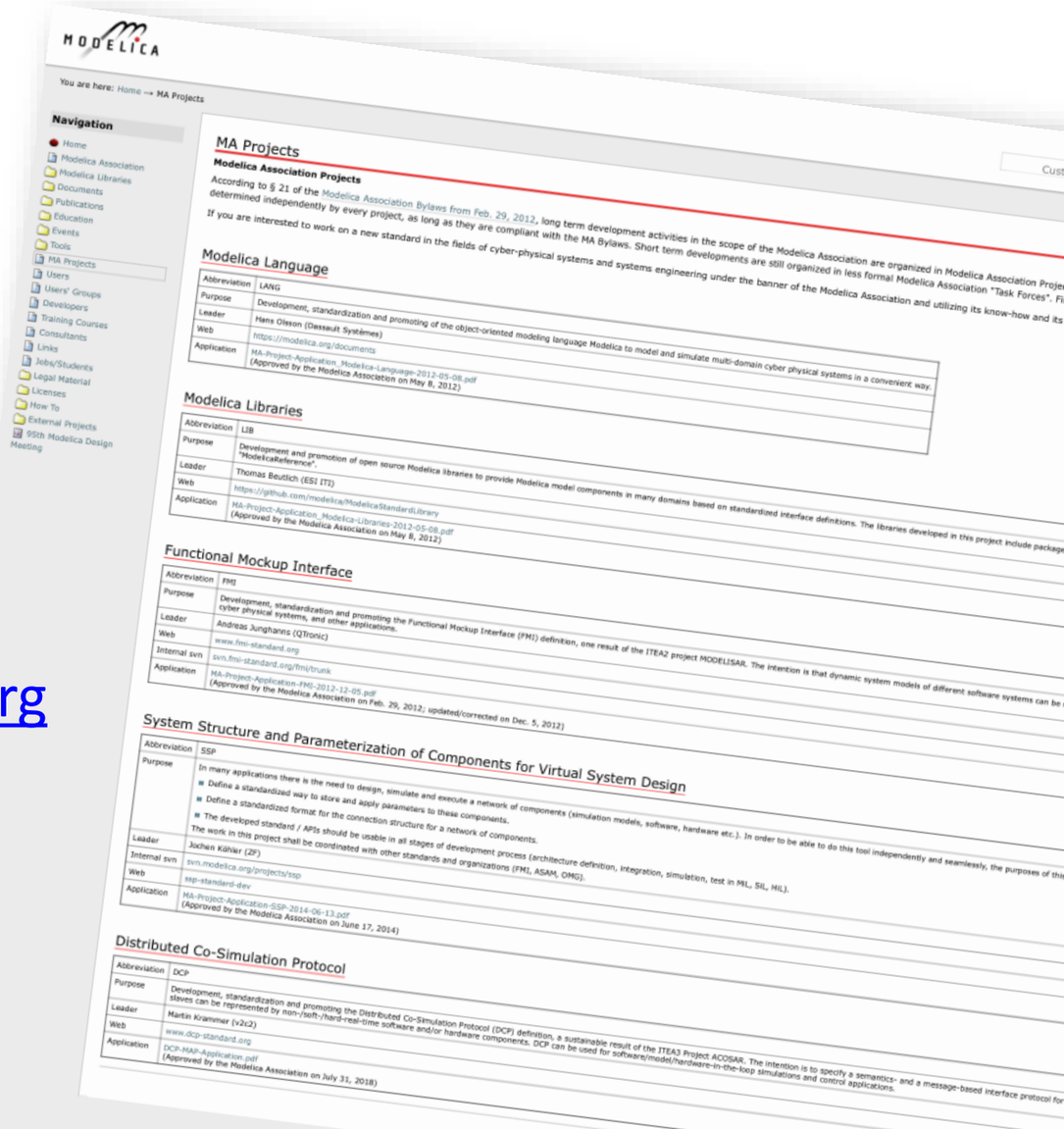


Setup on display at final event

Future of DCP

- The DCP 1.0-RC1 was submitted to **Modelica Association** for standardization
- Will be maintained as **Modelica Association Project (MAP)**
- Sustainable ACOSAR project result
  - The DCP will be freely available
  - Open for everyone!
- Website: www.dcp-standard.org

www.modelica.org

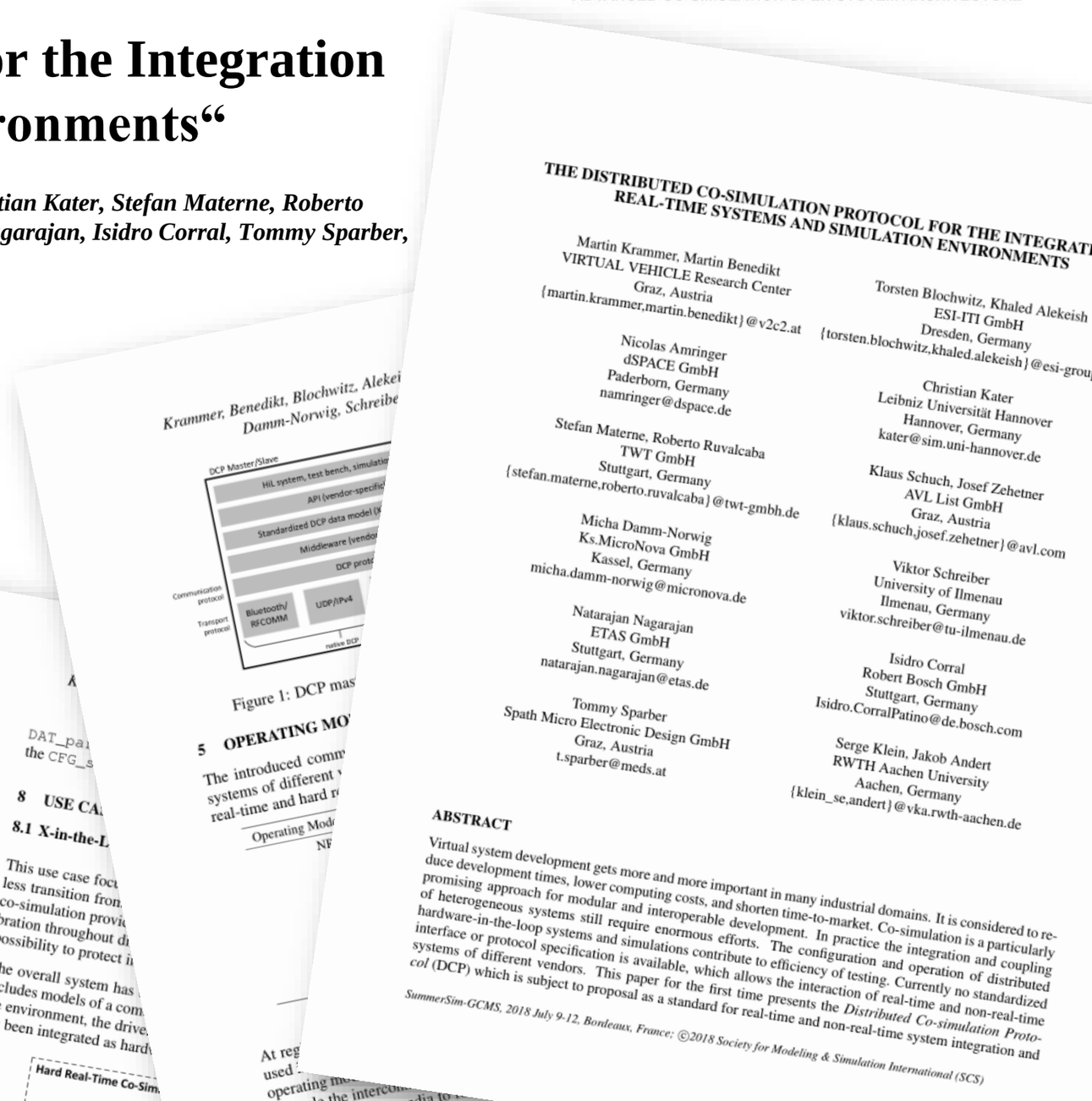


Publication

„The Distributed Co-Simulation Protocol for the Integration of Real-Time Systems and Simulation Environments“

Martin Krammer, Martin Benedikt, Torsten Blochwitz, Khaled Alekeish, Nicolas Amringer, Christian Kater, Stefan Materne, Roberto Ruvalcaba, Klaus Schuch, Josef Zehetner, Micha Damm-Norwig, Viktor Schreiber, Natarajan Nagarajan, Isidro Corral, Tommy Sparber, Serge Klein and Jakob Andert

- **“Requirements Engineering for Consensus-Oriented Written Technical Specifications”**
  Martin Krammer, Nadja Marko and Martin Benedikt, accepted for publication at 26th IEEE International Requirements Engineering Conference, August 20-24, Banff, Alberta, Canada
- **“Master for Simulation Control using the Distributed Co-Simulation Protocol”**
  Martin Krammer, Martin Benedikt, accepted for publication at IEEE 16th International Conference on Industrial Informatics, July 18-20, Porto, Portugal
- **“Configuration of Slaves Based on the Distributed Co-Simulation Protocol”**
  Martin Krammer, Martin Benedikt, accepted for publication at 23rd International Conference on Emerging Technologies and Factory Automation, September 4th - 7th, 2018, Torino, Italy


THE DISTRIBUTED CO-SIMULATION PROTOCOL FOR THE INTEGRATI
REAL-TIME SYSTEMS AND SIMULATION ENVIRONMENTS

Martin Krammer, Martin Benedikt
VIRTUAL VEHICLE Research Center
Graz, Austria
{martin.krammer,martin.benedikt}@v2c2.at

Torsten Blochwitz, Khaled Alekeish
ESI-ITI GmbH
Dresden, Germany
{torsten.blochwitz,khaled.alekeish}@esi-grou

Nicolas Amringer
dSPACE GmbH
Paderborn, Germany
namringer@dspace.de

Christian Kater
Leibniz Universität Hannover
Hannover, Germany
kater@sim.uni-hannover.de

Stefan Materne, Roberto Ruvalcaba
TWT GmbH
Stuttgart, Germany
{stefan.materne,roberto.ruvalcaba}@twt-gmbh.de

Klaus Schuch, Josef Zehetner
AVL List GmbH
Graz, Austria
{klaus.schuch,josef.zehetner}@avl.com

Micha Damm-Norwig
Ks.MicroNova GmbH
Kassel, Germany
micha.damm-norwig@micronova.de

Viktor Schreiber
University of Ilmenau
Ilmenau, Germany
viktor.schreiber@tu-ilmenau.de

Natarajan Nagarajan
ETAS GmbH
Stuttgart, Germany
natarajan.nagarajan@etas.de

Isidro Corral
Robert Bosch GmbH
Stuttgart, Germany
Isidro.CorralPatino@de.bosch.com

Tommy Sparber
Spath Micro Electronic Design GmbH
Graz, Austria
t.sparber@meds.at

Serge Klein, Jakob Andert
RWTH Aachen University
Aachen, Germany
{klein_se,andert}@vka.rwth-aachen.de

ABSTRACT

Virtual system development gets more and more important in many industrial domains. It is considered to re-
duce development times, lower computing costs, and shorten time-to-market. Co-simulation is a particularly
promising approach for modular and interoperable development. In practice the integration and coupling
of heterogeneous systems still require enormous efforts. The configuration and operation of distributed
hardware-in-the-loop systems and simulations contribute to efficiency of testing. Currently no standardized
interface or protocol specification is available, which allows the interaction of real-time and non-real-time
systems of different vendors. This paper for the first time presents the *Distributed Co-simulation Proto-
col* (DCP) which is subject to proposal as a standard for real-time and non-real-time system integration and

SummerSim-GCMS, 2018 July 9-12, Bordeaux, France;

ACOSAR
ADVANCED CO-SIMULATION OPEN SYSTEM ARCHITECTURE

Any questions?

Martin Krammer

martin.krammer@v2c2.at