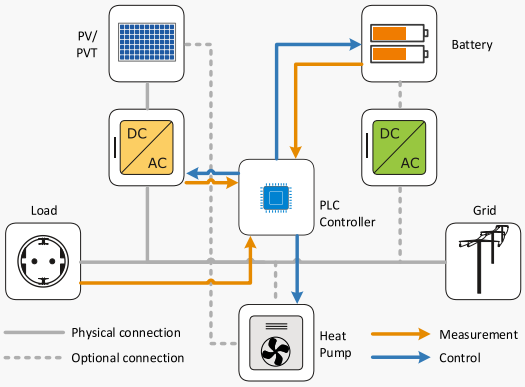
Development of a model-based control application compliant with IEC 61499 for building energy systems with a focus on photovoltaics

Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

Marc Jakobi[1] · Tjarko Tjaden[1] · Volker Quaschning[1] · Urs Stöckli[2] · Luc Meier[2]
[1]HTW Berlin · University of Applied Sciences · Wilhelminenhofstraße 75A · 12495 Berlin · Germany
[2]Vela Solaris AG · Stadthausstrasse 125 · 8400 Winterthur · Switzerland
jakobi@htw-berlin.de

System configurations



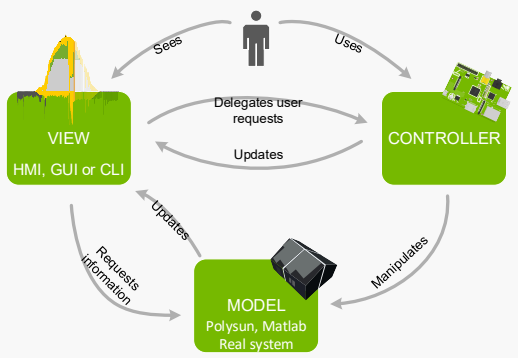
Motivation and objectives

Political marginal conditions for PV systems, such as feed-in limitations have resulted in the need for intelligent operation strategies.
Proprietary solutions available on the market today are costly and intelligent controllers for building energy systems can thus be classified as luxury products.
There is a need for generic software based on standards for the control of multi-generator systems. This contribution aims to provide an open source solution that can be used on a variety of inexpensive hardware.

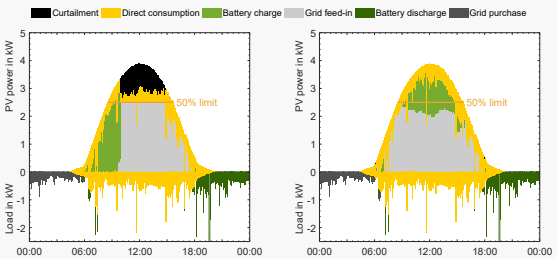
Objectives:

- **Development of communication libraries** that enable co-simulations between IEC 61499 control systems and simulation software (Polysun/Matlab)
- **Design and validation of an IEC 61499 control application** in 4diac using the communication libraries and simulation tools
- **Deployment of the application** and use in field tests
- **Decoupling of the controller** using the MVC design pattern
- **Establishment of an open source community** in the field of energy management

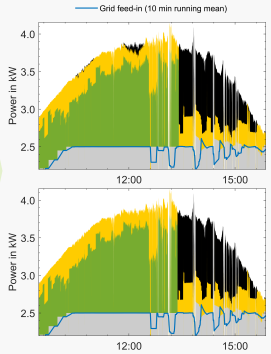
Model-View-Controller design pattern



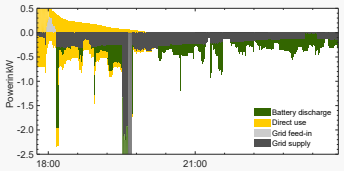
Matlab-4diac co-simulations: PVprog (forecast based battery operation) and curtailment



For optimal results, the PVprog algorithm needs to know the PV power before curtailment. This was achieved by incorporating both the PVprog and a PID controller for curtailment into the same application. If PVprog and curtailment are separated (Top right figure), they can interfere with each other. The results of a combined controller on the same day are depicted in the bottom right figure.



	Unit	No controller	PVprog (IEC 61499)	PVprog (Matlab)
Degree of self-sufficiency	%	53.6	52.7	52.6
Curtailment losses	%	6.8	1.4	0.9
Direct use	kWh	1,497.1	1,497.6	1,497.6
Battery charge	kWh	1,413.6	1,463.9	1,353.3
Battery discharge	kWh	1,186.6	1,144.9	1,136.0
Grid feed-in	kWh	1,769.1	2,090.8	2,123.4
Grid supply	kWh	2,325.9	2,367.6	2,376.5
Curtailment	kWh	349.1	68.0	46.1



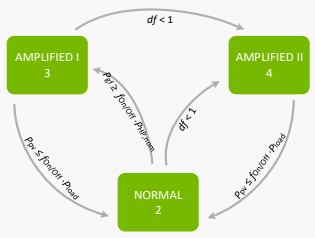
Optional feature:

Forecast-based load peak shaving can be used to avoid penalties and maximize battery use

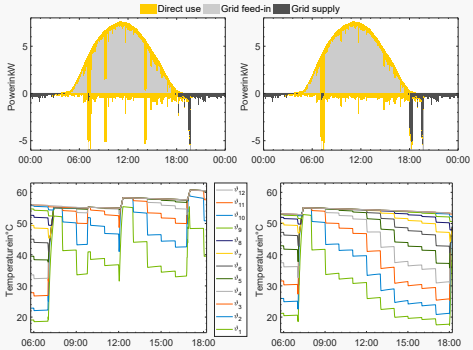
PV battery system:
- Left: Uncontrolled, simulated in Matlab
- Right: PVprog* - forecast based battery operation implemented in 4diac, co-simulated with Matlab

*PVprog was developed by the research group "Solar storage systems" at HTW Berlin.

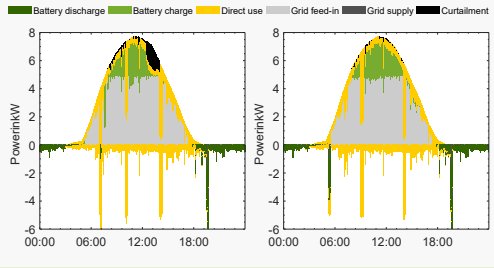
Polysun-4diac co-simulations: PVprog, curtailment and SG Ready heat pump DSM

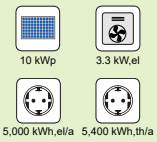


P_{pv}: PV power
$P_{HP,nom}$: Nominal heat pump power (electrical)
P_{load}: Load
P_{gf}: Feed-in power
$f_{on/off}$: Switching threshold (10 % - 150 %)
df: Derating factor (1 = no curtailment)



PV heat pump system:
- Left: With SG Ready controller. The drinking water tank is used to store excess PV energy, reducing the heat pump's use at night.
- Right: Without controller
- Bottom: Drinking water buffer tank temperatures per layer



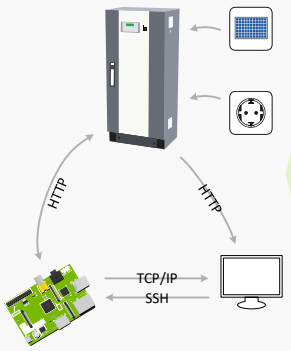


In a combined controller, interference of the PVprog and SG Ready control modes can be eliminated by measuring the heat pump's electrical load separately (above, right). This can be achieved by using smart components that comply with the EEBus "SPINE" protocol.

Hardware deployment: Initial field tests with a Raspberry Pi

Apart from the ability to design complex control systems, the main advantage of IEC 61499 compared to the current industry standard, IEC 61131-3, is platform independence.
IEC 61499 applications can easily be ported between integrated development environments (IDEs). Thus, they can be run on a large variety of hardware (a selection is listed below).

Hardware	Runtime Environemt	Source code
PC (Win32/Posix)	4diac-RTE	open source
Lego Mindstorms (EV3)	4diac-RTE	open source
Raspberry Pi / Raspberry Pi SPS	4diac-RTE	open source
Odroid	4diac-RTE	open source
Wago PFCs SPS	4diac-RTE	open source
ICnova	4diac-RTE	open source
µMic 200	4diac-RTE	open source
IndraControl XM22 PLC	4diac-RTE	open source
nxtDSCmini	by nxtControl	proprietary
CUB 100	by B-Control	proprietary
Any JAVA-based hardware	FBRT	closed source



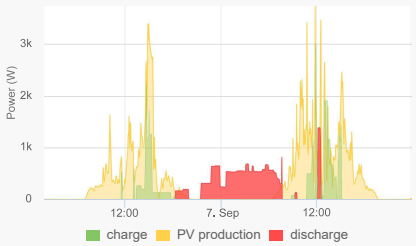
Exemplary setup:

- Raspberry Pi 2 (Raspbian OS) running 4diac-RTE
- 4.9 kWp PV system
- 5.3 kWh (usable) battery
- Battery hosts a RESTful server
- Communication via HTTP GET/PUT methods
- Apart from the communication protocol, the control application remains almost completely unchanged
- Monitoring via RESTful server or via controller

The following non-portable additions were made to the application for increased stability. To use them on a different RTE than 4diac, they must be ported over:

- Implemented a watchdog timer. If the Raspberry Pi's processor freezes or if 4diac-RTE stops running, the device is rebooted.
- Implemented the ability to save and load internal data upon reboot. This way, the forecasts, which can take up to 10 days to fully initialize, are not lost.

Preliminary results show a clear shift of the battery charging to times of higher PV production.



Summary and conclusion

- Open source communication libraries enable co-simulation of industry compatible IEC 61499 control applications with Polysun and Matlab
- Direct development of control applications in an IEC 61499 compliant IDE
- No prototyping necessary
- No porting from prototype to final product necessary
- The Libraries were used to develop an intelligent model-based control application for building energy systems

For optimal results...

- PVprog operation and PV curtailment should communicate (i.e. in a combined controller)
- DSM controlled load should be treated by PVprog as stored energy that reduces the load at a later time
- Use of intelligent devices (e.g., SPINE)

Outlook:

- Preliminary field testing proves to be very promising
- Implementation of SPINE communication protocol for 4diac planned
- In an ever-growing industry, the potential for further development of this project may never cease to exist

Source code

- **PVprog algorithm in Matlab:**
pvspeicher.htw-berlin.de/veroeffentlichungen/daten/pvprog/
- **tcpip4diac: Matlab - IEC 61499 communication library**
github.com/MrcJkb/tcpip4diac/
- **Polysun4diac: Polysun - IEC 61499 communication plugin**
github.com/MrcJkb/Polysun-4diac-ControllerPlugin/
- **IEC 61499 function block library + control application:**
github.com/MrcJkb/PVTControllerLib/
- **HTTP communication layer for 4diac-RTE:**
github.com/MrcJkb/forte_http_comm/
- **EEBus "SPINE" communication layer for 4diac-RTE:**
(development will begin shortly)
github.com/MrcJkb/forte_spine_comm/