

## Topics:

- 1. Network Processes via a Esona Server**
- 2. Control Panels of the devices**
- 3. The CSBP Group Object**
- 4. Object Net's, first steps**

# Messhütte

## Server

### CS

- Interface Objekte
- Ausfall Kontrolle

OPC Server

## Linux-Client

### CS

- Benutzer GUI's  
(Eingabepanels)

## WinXP-Client

### CS

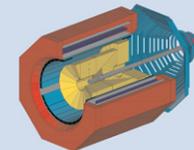
- Benutzer GUI's  
(Eingabepanels)

...

# TCP/IP

## CAVE

PT100 Temp.Sensoren  
Beckhoff BC9000



Esome Server

## CAMAC BUS

CAEN- Hoch  
Spannungs-  
versorgung  
-18 Channels

LeCroy – HV  
Devices  
-1792  
Channels

Constant  
Fractions,  
ca. 150 Stck.

Ethernet – GBIP  
Controller

## GPiB-Bus

Lambda, low-Voltage device

Lambda, low-Voltage device

...

# kp1pc065 (FOPI CS Server)

## CS

### Esome Dispatcher

- use the Labview Esome Driver to communicate with the Server
- only the commands cfsa and cdreg seems to work

### CS Events

### LeCroy 2132 Interface

- can read and write Voltages, etc. from a LeCroy 1440 Module

### Constant Fraction Interface

- can read and write values to CF8101 modules

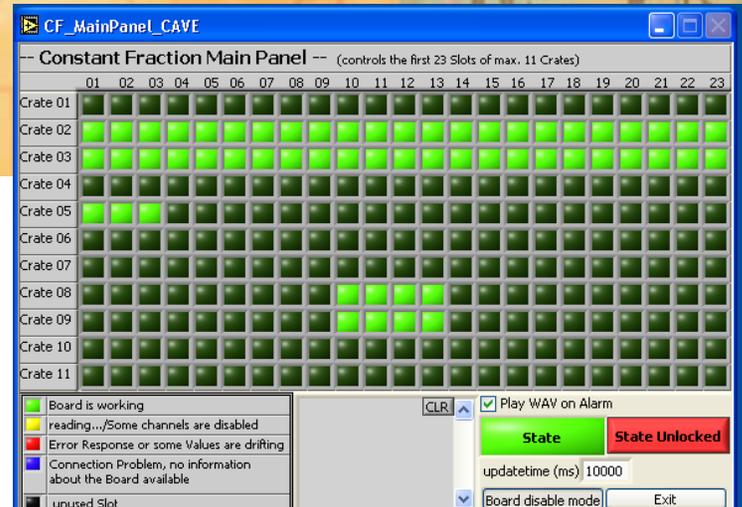
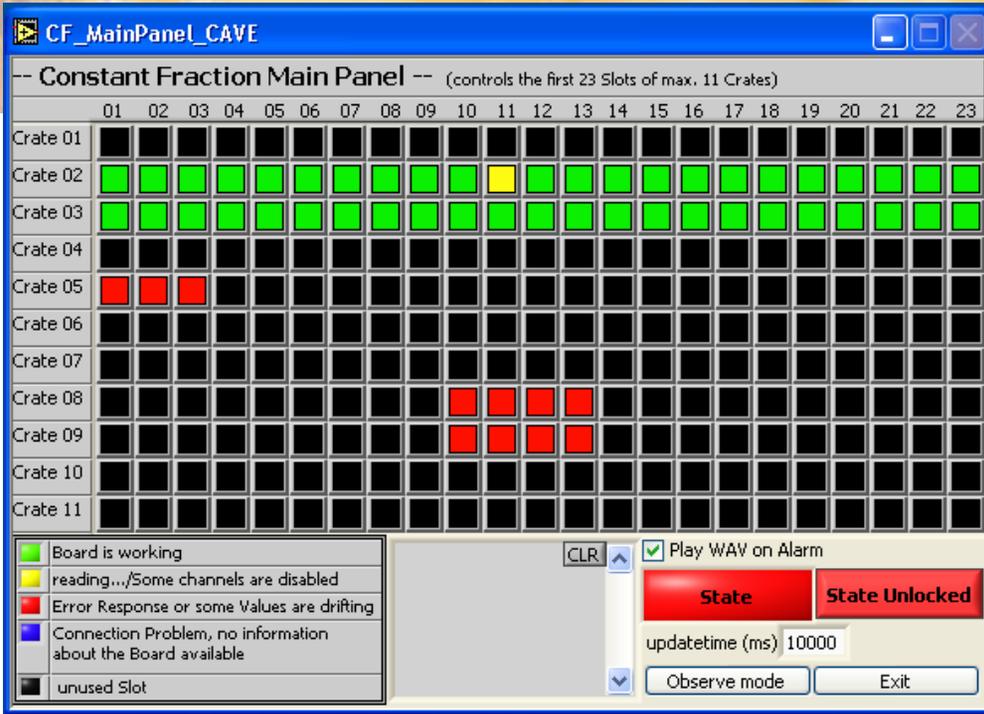
**TCP/IP**

**Esome Server**

**Camac Bus**

- Esome Dispatcher provides single CAMAC CNAF's from the Labview Event, to the real CNAF
- Esome driver is coded in C and only fitted in as ".dll" by a call Library node
- To talk with the esome dispatcher only send the event, that fits your camac-action, for example cfsa

# Constant Fraction's



- Control of ca. 150 Constant Fraction "CF 8101" Discriminators

- control via Esone Server on a CAMAC Bus

- CNAF's are the CAMAC Data Words (Crate, Board, Address, Function)

- Able to observe all values, and to recognize chances

- Enable/Disable Boards to observe

- Able to play a alarm-wav file on alarm



# LeCroy HV

- Old Hardware (1985)
- control via Esone Server on a CAMAC Bus
- Very Slow reaction time
- Output is written in Output Buffer of the 2132 Interface
- Device Language is 16 Bit-Codet and Cryptic
- Control of more than 1700 HV-Channels

LeCroy1440\_MainPanel

File Edit Operate Tools Browse Window Help

-- LeCroy1440 Main Panel -- (controls up to 7 LeCroy1440 Mainframes via a LeCroy 2132 Interface Module)

HV State [ON/OFF]	Board																pos. Curr. Limit [mA]		neg. Curr. Limit [mA]			
	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	Demand	Value	Demand	Value		
Mainframe 1																	0	Set	0	0	Set	0
Mainframe 2																	2	Set	2	0	Set	2,45
Mainframe 3																	2,3	Set	2,3	2,3	Set	2,3
Mainframe 4																	0	Set	0	0	Set	0
Mainframe 5																	0	Set	0	0	Set	0
Mainframe 6																	0	Set	0	0	Set	0
Mainframe 7																	0	Set	0	0	Set	0

Legend:  
Green: Board is working  
Yellow: reading...  
Red: Error Response or some Values are drifting  
Blue: Connection Problem, no information about the Board available  
Black: unused Slot

Buttons: CLEAR, State, State Unlocked, Observe mode, Exit

Play WAV on Alarm  updatetime [sec] 30

LeCroy1440-Module\_2\_11

File Edit Operate Tools Browse Window Help

Channel	Demand [V]	Set	actual Value [V]
Channel 01	0	Set	-8
Channel 02	0	Set	-3
Channel 03	0	Set	-8
Channel 04	0	Set	-18
Channel 05	0	Set	-8
Channel 06	0	Set	-8
Channel 07	0	Set	-8
Channel 08	0	Set	-8
Channel 09	0	Set	-8
Channel 10	0	Set	-8
Channel 11	0	Set	-8
Channel 12	0	Set	-8
Channel 13	0	Set	-9
Channel 14	0	Set	-8
Channel 15	0	Set	-8
Channel 16	0	Set	-8

Buttons: State, Exit, Mainframe HV On, Fast Set, Voltage -0, Set to Board, Set to Mainframe, Write Values, Read Values

LeCroy1440-Module\_2\_11

File Edit Operate Tools Browse Window Help

State Exit

Mainframe HV On  
Mainframe 2  
Board 11

Fast Set  
Voltage -0  
Set to Board  
Set to Mainframe

Write Values  
Read Values



# Low Voltage



- Control of up to 5 Lambda-Genesys devices via GPIB Bus

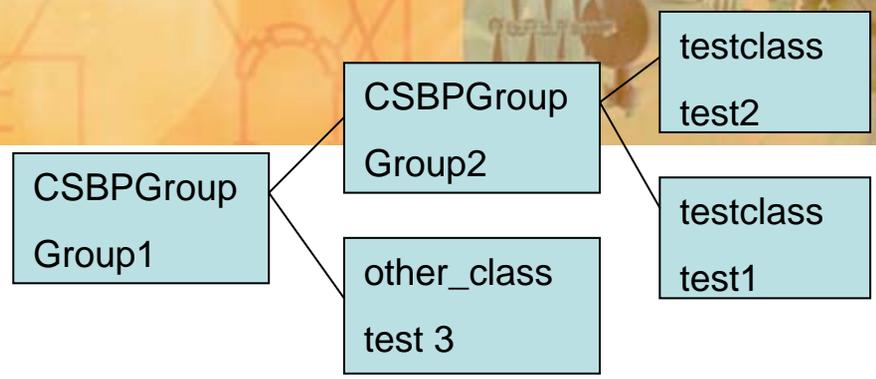
- Use of the device-specific drivers

- possibility to save all changes, done to the channels into a logfile



# The CSBP Group Object

- is for grouping many BaseProcess-Classes
- to make talking to them more easy
- to give some structure to your Project
- and to get rid of the loadProcess procedure



```
addObjects |vmsg:/"#SELF#"/addObjects [T]
deleteObjects |vmsg:/"#SELF#"/deleteObjects [T]
getState |vmsg:/"#SELF#"/getState [T]
deleteGroup |vmsg:/"#SELF#"/deleteGroup [T]
```

```
addObjects * n[Array of Objects(string)] //adds the Objects
in the list to the Group

deleteObjects * n[Array of Objects(string)] //deletes the
Objects in the list from the Group

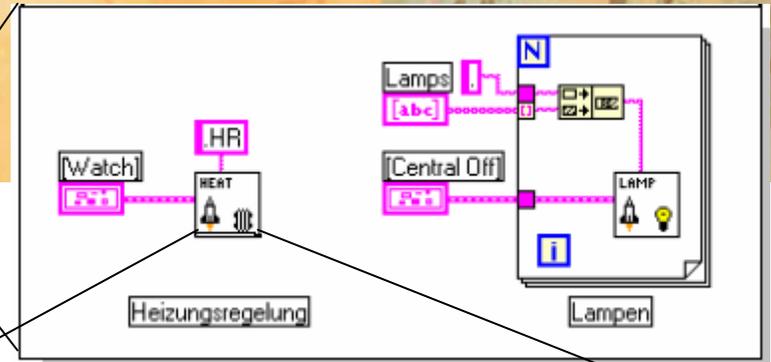
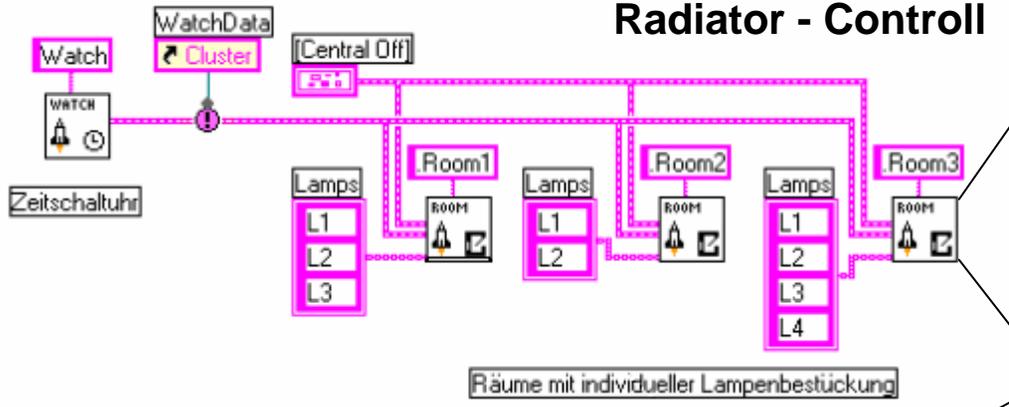
getState (s,*) n[(Objectclassname,Array of Objects)] //
returns the state of CSBPGroup

deleteGroup b n[delete Elements too?]/Deletes the Group,
and if the bool is true all its elements too
```

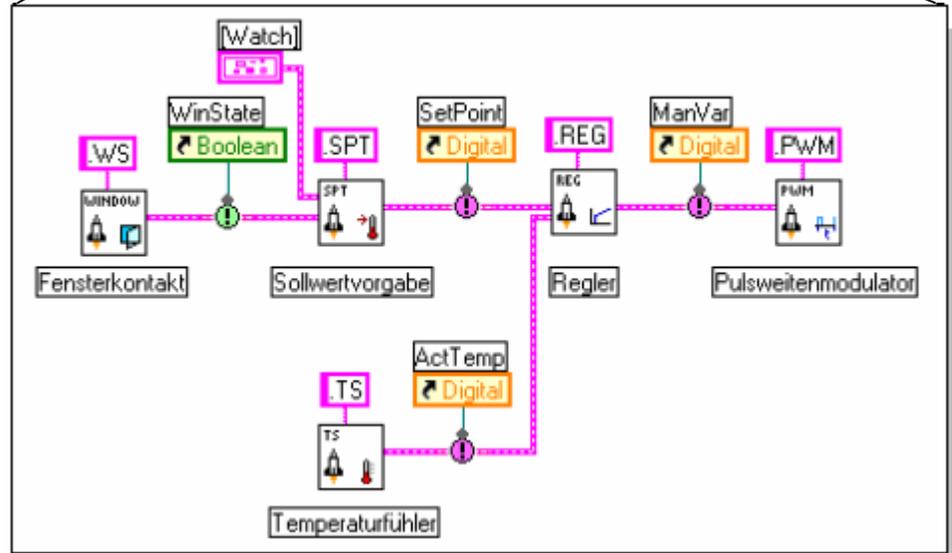
e following:  
r (that is the selector that will be used later when an event has arrived)  
at specifies not only the name of the URL but also the protocol used.  
to enable/disable the event.

- "get State" gets also the state of eventually sub Groups, needs no argument
- if you send a event to the GroupObject, it will provide it to all its members.
- The data array in the answer cluster has the laengt of the number of responses, made by the member objects
- you can also create group elements on other nodenames, if the CS there knows the class
- if you set the bool in "deleteGroup" the whole Group-tree will also get deleted

## Radiator - Controll



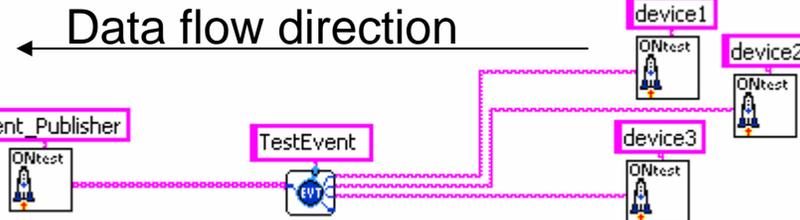
## Object Net, made with ObjectView

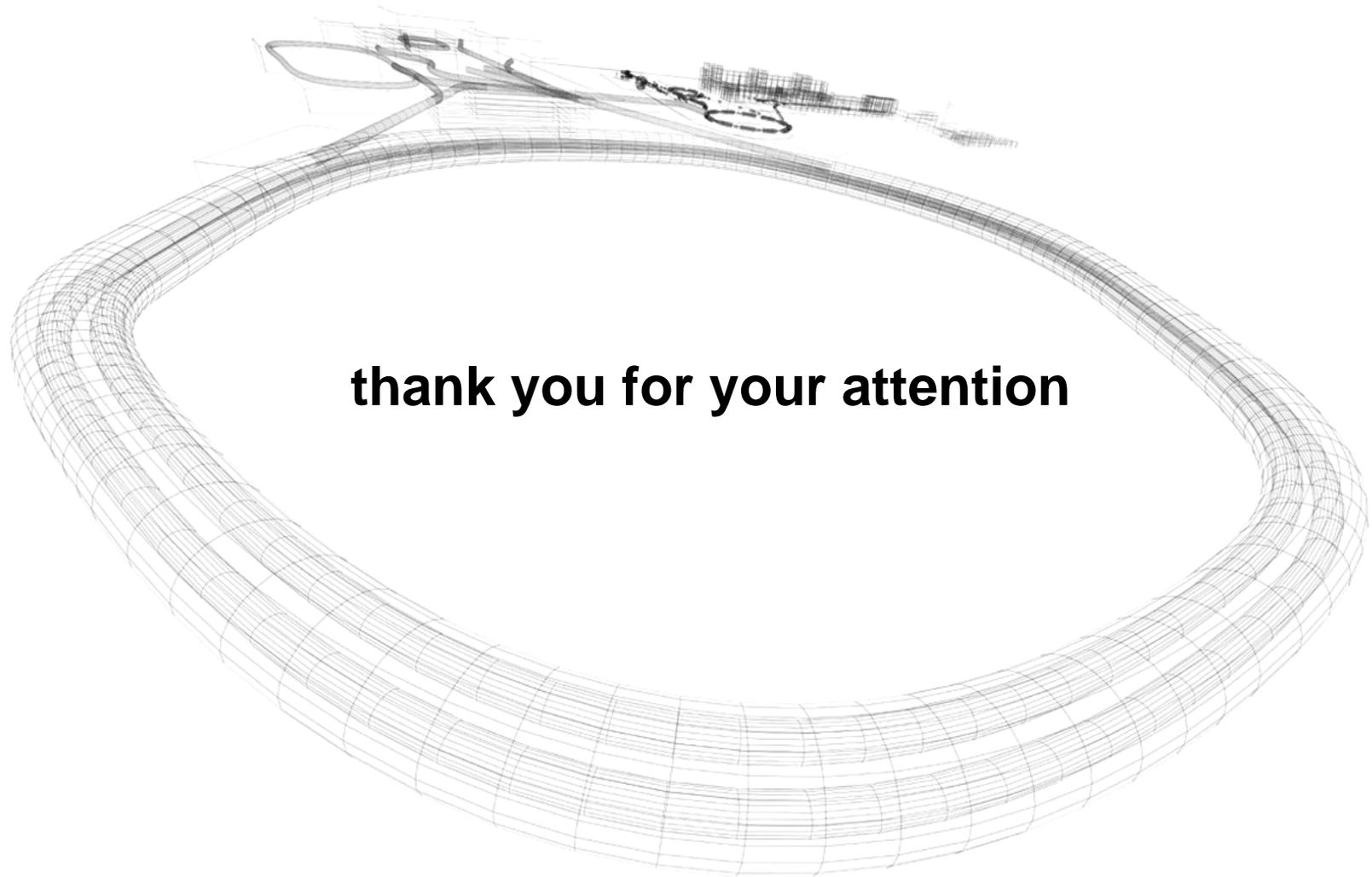


## Advantages of Object Nets

- the whole Project on one Page
- a startup procedure is no longer nesessary
- You can "see" which Objects are talking together
- also "LoadProcess" on other Machnines could be implemented, to produce a distributed Object Network

## Implementation in the CS ?





**thank you for your attention**