CS++ - Libraries extending the NI Actor Framework

The native LVOOP based successor of the *CS* Framework

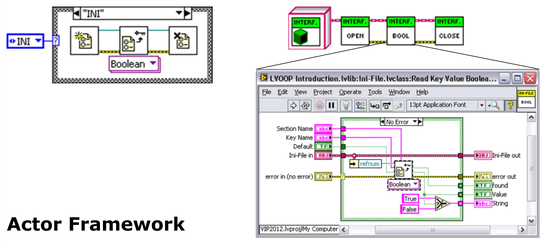
H.Brand, D.Neidherr, EEL, 2.12.2019

Purpose: **CS++** libraries ease development and commisioning of heterogenious distributed experiment control systems.

CS++ extends **AF** with missing functionalities which are need in the experimenta physics domain where experiments are dynamically developed, rearranged and extended on the fly. **CS++** adds Factory with configuration database, Process Variables, After-Launch-Init, Pseudo-Periodic Polling, Settings, Asynchonous Callback Messages, Central Messge Logging, Data Logging, Introspection, ObjectManager, StartActor, which can automatically (re-)lauch nested actors by configuration, and a custom CS++MessageMaker that supports the generation **CS++** Messages using VI-Scripting.

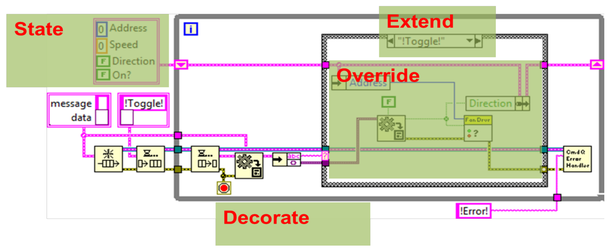
LVOOP: Possible cases for the application of classes

- Cluster or type definitions can be replaced with classes.
- Derives classes add attributes to the ancestor class.
- Replacement of data type dependent case-structures by dynamic dispatching.
- Dependent of the objects class the correct corresponding Override-VI is called.



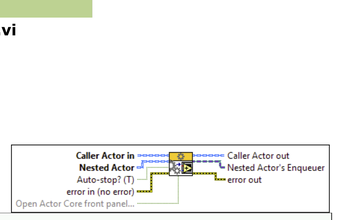
Actor Framework

The application layer uses base classes only. Details are implemented in derived classes.

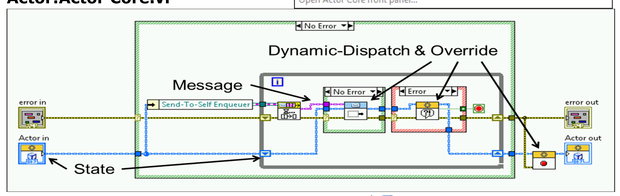


QDMH.vi → **Actor Core.vi**

State Cluster	→ Actor Class
Command Cluster	→ Message Class
Case Structure	→ Message:Do.vi
Error-Handling	→ Handle Error.vi
Stop	→ Stop Core.vi

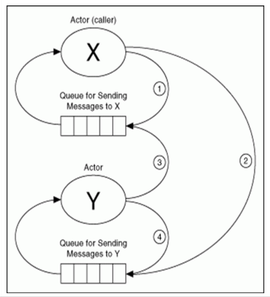


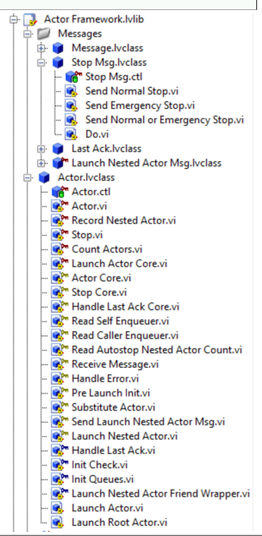
Actor:Actor Core.vi



Local Actor Communication

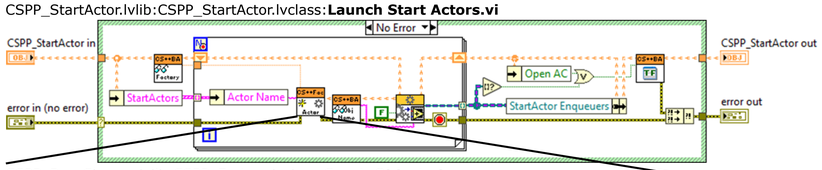
- Each Actor has a Message-Queue.
- Message classes are the public interface.
- Communication pathes:
- Actor to Self (1,4)
- Caller-Actor to Nested-Actor (2)
- Nested-Actor to Caller (3)



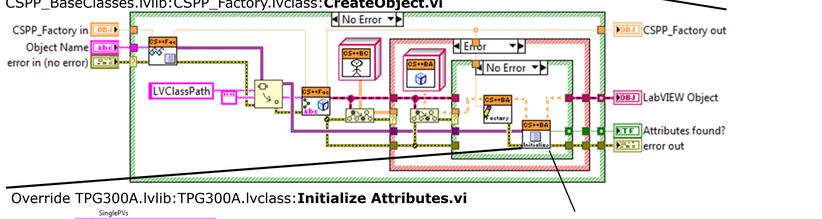


StartActor, Factory & Object Initialisation

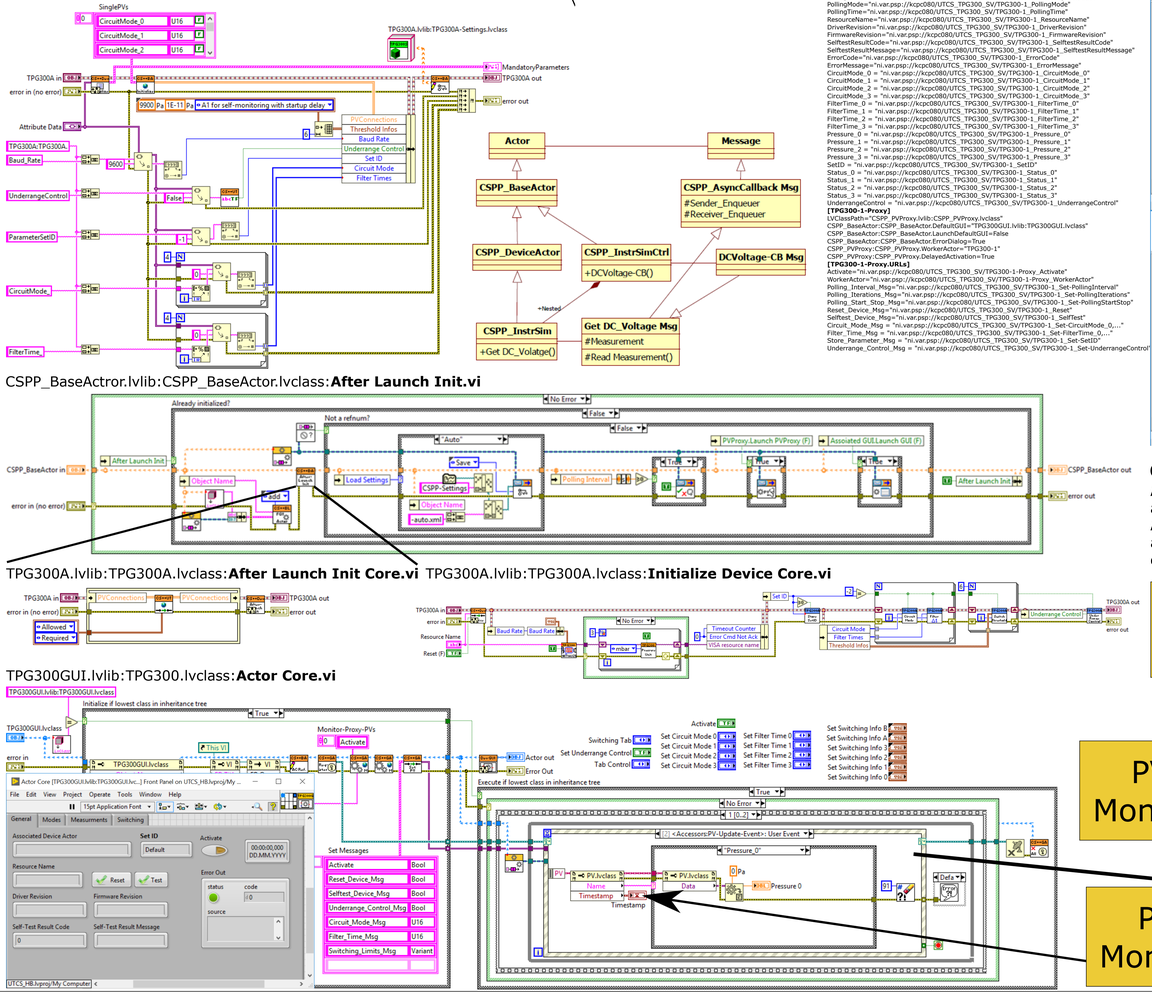
CSPP_StartActor.lvlib:CSPP_StartActor.lvclass:**Launch Start Actors.vi**



CSPP_BaseClasses.lvlib:CSPP_Factory.lvclass:**CreateObject.vi**



Override TPG300A.lvlib:TPG300A.lvclass:**Initialize Attributes.vi**




```
[StartActor]
LVClassPath="CSPP_StartActor.lvlib:CSPP_StartActor.lvclass"
CSPP_BaseActor::CSPP_BaseActor.PollingInterval_s=1.
CSPP_StartActor::CSPP_StartActor.Open_ActorCore=False
CSPP_BaseActor::CSPP_BaseActor.ErrorDialog=True
CSPP_StartActor::CSPP_StartActor.MessageLogger="DSClog"
CSPP_StartActor::CSPP_StartActor.StartActors=StartActor.StartActors
CSPP_StartActor::CSPP_StartActor.ActorList=StartActor.ActorList
[StartActor.StartActors]
#Actor objects to start with option to relaunch.
1UTCS_MainGUI=False
TPG300-1=True
[StartActor.ActorList]
#List of Actor object to be started manually
DSCAlarmViewer=False
DSCManager=False
DSCTrendViewer=False
TPG300-1=False
[1UTCS_MainGUI]
LVClassPath="UTCS_MainGUI.lvlib:UTCS_MainGUI.lvclass"
CSPP_BaseActor::CSPP_BaseActor.ErrorDialog=True
UTCS_MainGUI::UTCS_MainGUI.BMIL="BMIL"
UTCS_MainGUI::UTCS_MainGUI.Alarms="DSCAlarmViewer"
[TPG300-1]
LVClassPath="TPG300A.lvlib:TPG300A.lvclass"
CSPP_BaseActor::CSPP_BaseActor.Settings="Ignore"
CSPP_BaseActor::CSPP_BaseActor.DefaultGUI="TPG300GUI.lvlib:TPG300GUI.lvclass"
CSPP_BaseActor::CSPP_BaseActor.LaunchDefaultGUI=False
CSPP_BaseActor::CSPP_BaseActor.ErrorDialog=False
CSPP_BaseActor::CSPP_BaseActor.PollingInterval_s=0.
CSPP_BaseActor::CSPP_BaseActor.PVProxy="TPG300-1-Proxy"
CSPP_BaseActor::CSPP_BaseActor.LaunchPVProxy=True
CSPP_DeviceActors::CSPP_DeviceActor.ResourceName="TCPIP0::kp2com001.gsi.de::8000::SOCKET"
CSPP_DeviceActors::CSPP_DeviceActor.Reset=False
CSPP_DeviceActors::CSPP_DeviceActor.Selftest=False
CSPP_DeviceActors::CSPP_DeviceActor.ResetWithDefaults=False
TPG300A::TPG300A.Baud_Rate = 9600
TPG300A::TPG300A.ParameterSetID = -1
#Filter 0:Fast, 1:Medium, 2:Slow
TPG300A::TPG300A.CircuitMode_0 = 0
TPG300A::TPG300A.CircuitMode_1 = 0
TPG300A::TPG300A.CircuitMode_2 = 0
TPG300A::TPG300A.CircuitMode_3 = 0
TPG300A::TPG300A.FilterTime_0 = 0
TPG300A::TPG300A.FilterTime_1 = 0
TPG300A::TPG300A.FilterTime_2 = 0
TPG300A::TPG300A.FilterTime_3 = 0
TPG300A::TPG300A.UnderrangeControl = False
[TPG300-1.URLs]
PollingCounter="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_PollingCounter"
PollingDeltaT="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_PollingDeltaT"
PollingInterval="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_PollingInterval"
PollingIterations="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_PollingIterations"
PollingMode="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_PollingMode"
PollingTime="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_PollingTime"
ResourceName="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_ResourceName"
DriverRevision="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_DriverRevision"
FirmwareRevision="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_FirmwareRevision"
SelftestResultCode="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_SelftestResultCode"
SelftestResultMessage="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_SelftestResultMessage"
ErrorCode="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_ErrorCode"
ErrorMessage="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_ErrorMessage"
CircuitMode_0 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_CircuitMode_0"
CircuitMode_1 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_CircuitMode_1"
CircuitMode_2 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_CircuitMode_2"
CircuitMode_3 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_CircuitMode_3"
FilterTime_0 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_FilterTime_0"
FilterTime_1 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_FilterTime_1"
FilterTime_2 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_FilterTime_2"
FilterTime_3 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_FilterTime_3"
Pressure_0 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Pressure_0"
Pressure_1 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Pressure_1"
Pressure_2 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Pressure_2"
Pressure_3 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Pressure_3"
SetID = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_SetID"
Status_0 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Status_0"
Status_1 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Status_1"
Status_2 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Status_2"
Status_3 = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Status_3"
UnderrangeControl = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_UnderrangeControl"
[TPG300-1-Proxy]
LVClassPath="CSPP_PVProxy.lvlib:CSPP_PVProxy.lvclass"
CSPP_BaseActor::CSPP_BaseActor.DefaultGUI="TPG300GUI.lvlib:TPG300GUI.lvclass"
CSPP_BaseActor::CSPP_BaseActor.LaunchDefaultGUI=False
CSPP_BaseActor::CSPP_BaseActor.ErrorDialog=True
CSPP_PVProxy::CSPP_PVProxy.WorkerActor="TPG300-1"
CSPP_PVProxy::CSPP_PVProxy.DelayedActivation=True
[TPG300-1-Proxy.URLs]
Activate="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1-Proxy_Activate"
WorkerActor="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1-Proxy_WorkerActor"
Polling_Interval_Msg="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Set-PollingInterval"
Polling_Iterations_Msg="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Set-PollingIterations"
Polling_Start_Stop_Msg="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Set-PollingStartStop"
Reset_Device_Msg="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Reset"
Selftest_Device_Msg="ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_SelfTest"
Circuit_Mode_Msg = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Set-CircuitMode_0;..."
Filter_Time_Msg = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Set-FilterTime_0;..."
Store_Parameter_Msg = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Set-SetID"
Underrange_Control_Msg = "ni.var.psp://kcpc080/UTCS_TPG300_SV/TPG300-1_Set-UnderrangeControl"
```


CSPP_BaseActor.lvlib:CSPP_BaseActor.lvclass:**After Launch Init.vi**

TPG300A.lvlib:TPG300A.lvclass:**After Launch Init Core.vi**

TPG300A.lvlib:TPG300A.lvclass:**Initialize Device Core.vi**

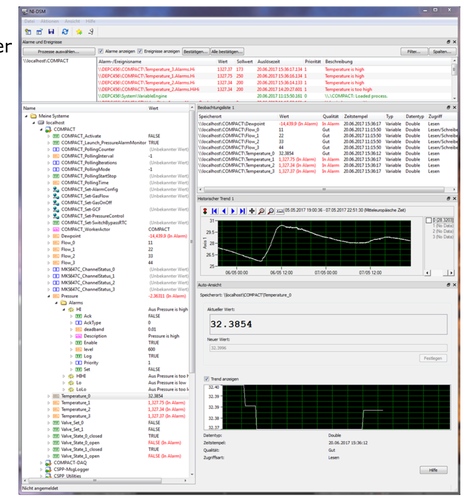
TPG300GUI.lvlib:TPG300.lvclass:**Actor Core.vi**

NI Tools

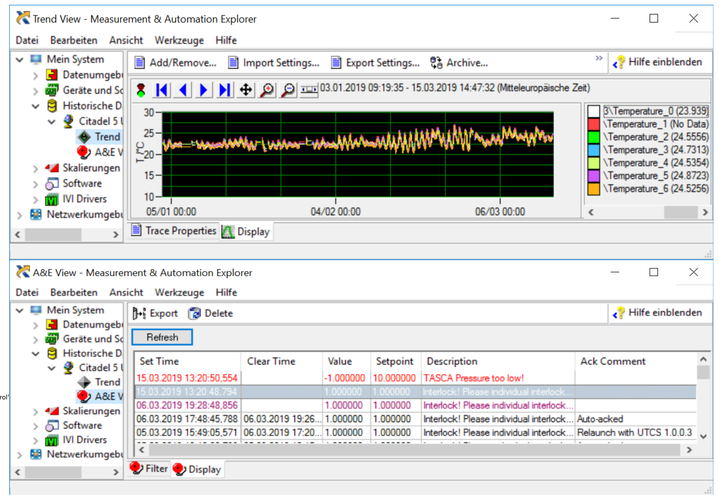
Distributed System Manager

Features:

- Alarm & Event Monitoring
- Distributed Process List
- Process
- Shared Variable List
- Selected SV List
- Historical Data Trending
- Online Monitor/Trending
- Processes, (Bound-)SV,
- Datalogging and Alarming
  can be configured from DSM.
- **Workspace: load/save**
- Sections can be un-/docked
- Very useful for adhoc
  monitoring and debugging
  without dedicated GUI.



Measurement & Automation Explorer



Coupling of Actor, Message and Process Variable

All CS++ actors publish their state to process variables, shared variables by default. Any other CS++ actor can launch a PVMonitor and register messages to be dispatched on value change or alarm/event. A GUI can monitor PVs for display, other actors may react to PV set-values. PVProxy is a class that allows to associate actor messages with PV-URLs by configuration. CS++MessageMaker decorates derived actor messages with PV parsing.