Laser Control System Workshop@EMIS

CS++ Add-On Libraries for the NI Actor Framework

H.Brand, D.Neidherr

Motivation I
CS Framework

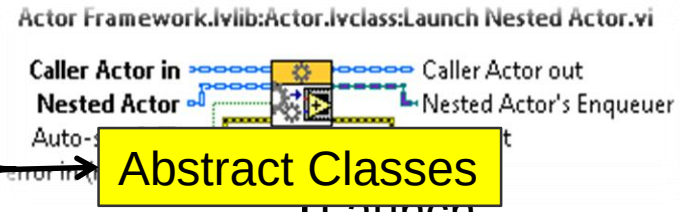
- *CS* Framework
  - *CS* is a LabVIEW/DIM based control system framework
    - multi-threaded,
    - event driven,
    - object oriented and
    - distributed with
    - SCADA functionality.
  - An experiment control system can be developed by combining the *CS* framework with experiment specific add-ons.
  - *CS* is supported on MS-Windows and on Linux (real-time OS Pharlap, LabVIEW RT)
- Artificial object-oriented approach started with LabVIEW 6i
  - Reference based (VI-Server), Multiple Inheritance like C++
  - *Complex with many recommendations which cannot be enforced.*
- Network layer: **D**istributed **I**nformation **M**anagement (DIM)
- Mainly used with Laser (PHELIX, POLARIS) and many Iontraps

Motivation II
LVOOP - Actor Framework – NI Tools

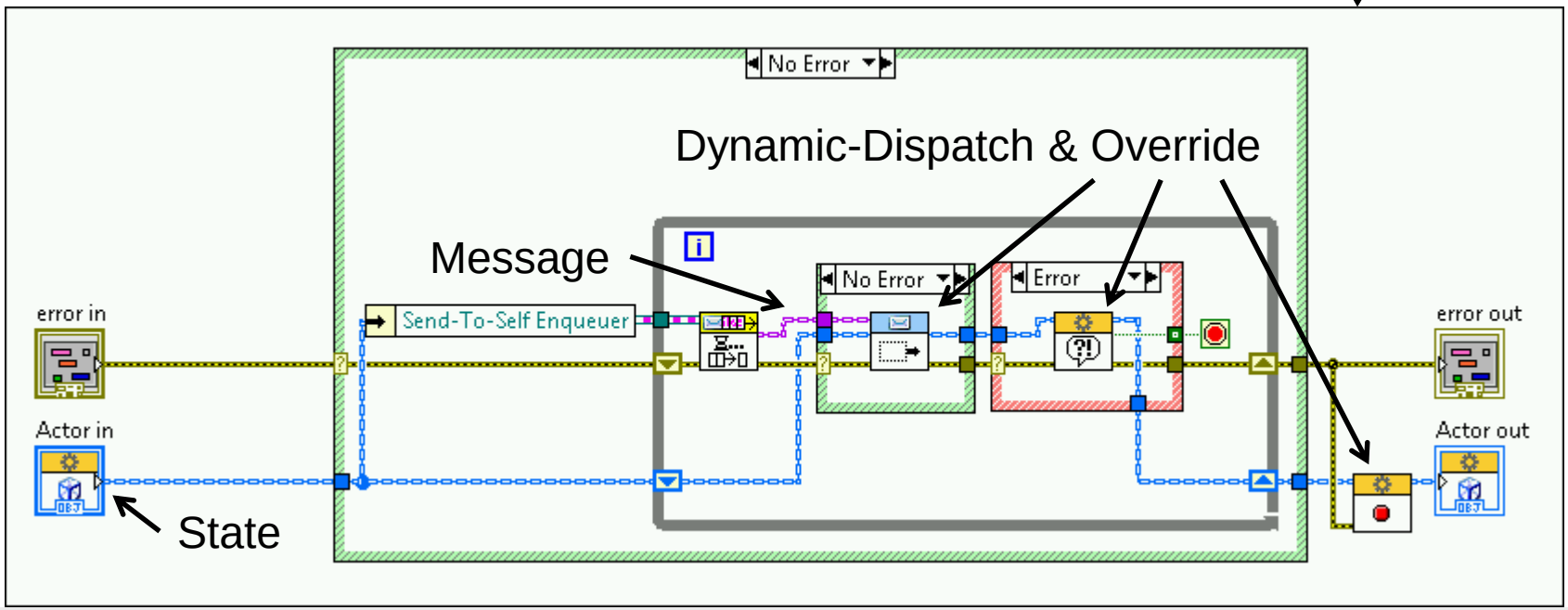
- Successful Feasibility Study: Mobile Agent
  - Based on LVOOP (LabVIEW 8.5)
  - Diploma Thesis
- NI Actor Framework provides simple and efficient design
  - Released with LV 2012
  - First application: Gas Flow Control for the COMPACT Detector
- Profit from NI maintenance and community developments
- Integrate non LabVIEW experts like short term Bachelor & Master students
- CLAD level implementing derived classes using a cooking recipe
- Use as much NI Tools as possible
  - Data Logging & Supervisory Control Module (DSC)
  - Distributed System Manager (DSM)
  - TDMS & DIAdem
  - (Teststand)

QSM → Actor Framework

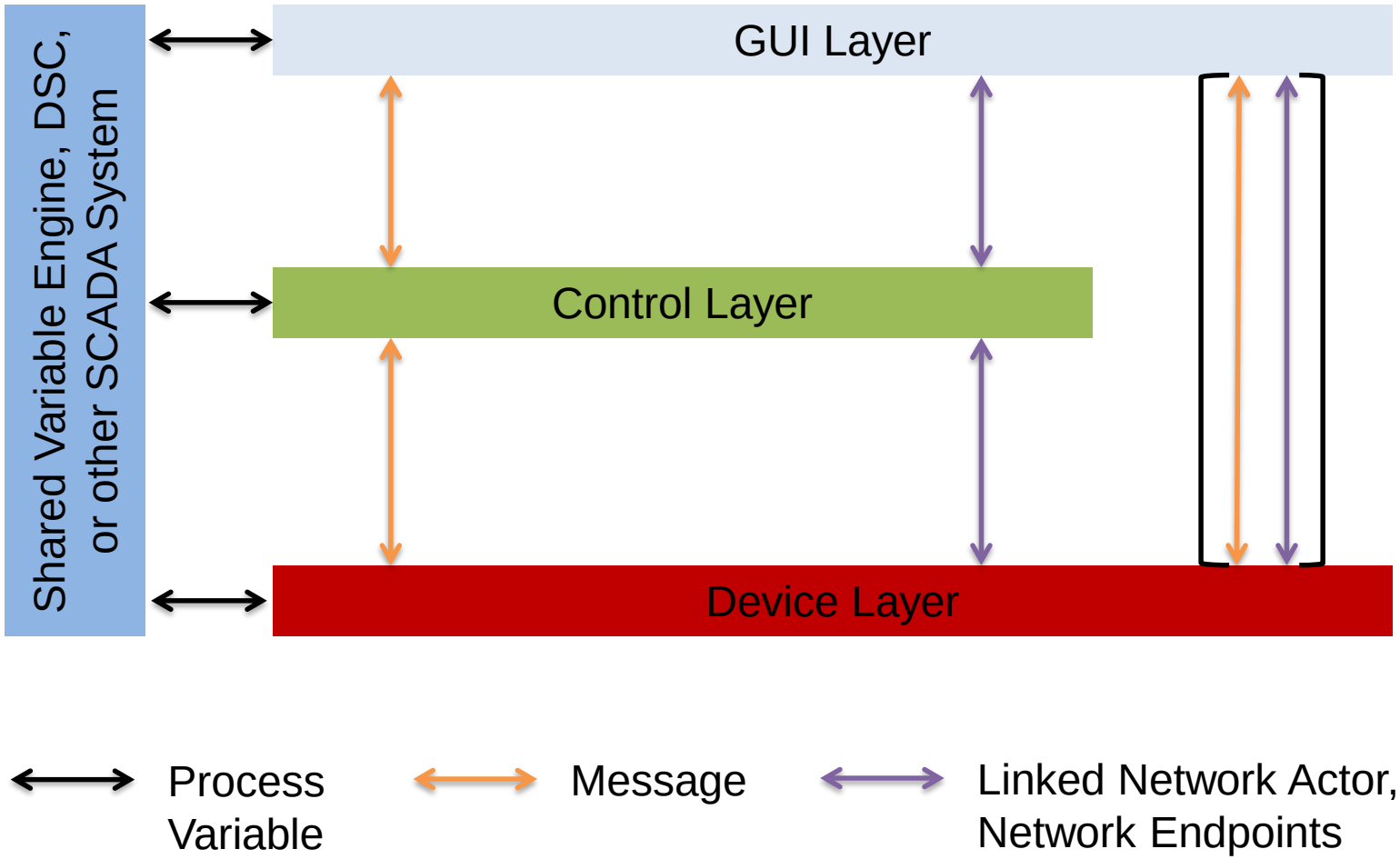
- QSM.vi → Actor Core.vi
  - State Cluster → Actor Class
  - Command Cluster → Message Class
  - Case Structure → Message:Do.vi
  - Error-Handling → Handle Error.vi
  - Stop → Stop Core.vi



Launch
(uses VI-Server)

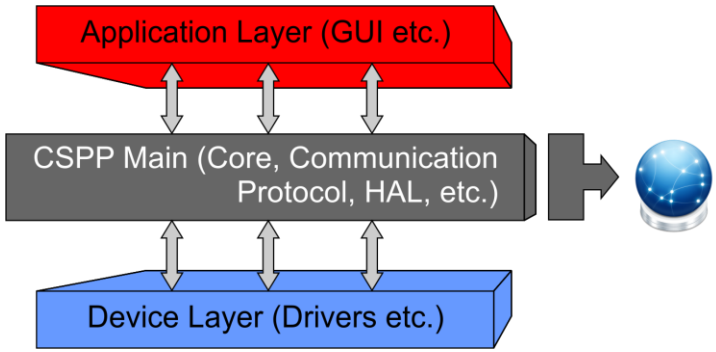


CS++ Architecture & Communication



Migration I
CS Framework → CS++

- **Similar feature set** as *CS* Framework based on
  - LVOOP & Dataflow
  - Actor Framework
- **Plan**: Extent Actor Framework
  - Concentration on necessary features for our Control Systems!
- **Architecture**: 3-Layer



- **Developer groups**
  - Core Developer:
    - **CS++** core and add-on libraries
    - Profound LVOOP/AF and **CS++** knowledge
  - Local Developer:
    - experiment-specific GUI- and Device-Actors
    - ~CLAD knowledge sufficient
  - User:
    - Use and/or configuration of Application

CS++ Status & Outlook

- Focus of development
  - Development of hardware layer
  - Commissioning of hardware layer
  - Simple Device GUI‘s
  - Stability of core system
  - Device Base Classes
  - Generic interface classes
- Projects & Applications
  - Motion Control (CaveA)
  - TASCA and COMPACT (LabVIEW-FPGA)
  - Under development
    - GEMDiscProduction (LabVIEW-RT-FPGA)
    - Vacuum Heating Control (LabVIEW-RT)
  - Planned developments
    - Migrate SHIPTRAP/HITRAP/TRIGATRAP
      - New Sequencer
      - Interface to MMn
    - Laser Spectroscopy at GSI
    - MATS for FAIR
- CS++Tools
  - CS++MessageMaker continuous improvements
  - CS++Configuration under development

References

- DIM: https://dim.web.cern.ch/dim/
- Actor Framework
  - https://forums.ni.com/t5/Actor-Framework-Documents/tkb-p/7301
- CS++
  - H.Brand, D. Neidherr, Scientific Report 2014 GSI Report 2015-1, 459 p. (2015), http://repository.gsi.de/record/184173/files/FG-GENERAL-41.pdf
  - H.Brand, D. Neidherr; “CS++ - NI Actor Framework-based Class Library”; "Virtuelle Instrumente in der Praxis 2016 - Begleitband zum 21. VIP-Kongress", Rahman Jamal, Ronald Heinze (Hrsg.), VDE Verlag, ISBN 978-3-8007-4208-0
  - H.Brand, D. Neidherr, D. Krebs, B.Voss; “Anwendung von AF/CS++ auf CompactRIO am Beispiel der GEM-Disc-Produktionsanlage für PANDA@FAIR”; "Virtuelle Instrumente in der Praxis 2017 - Begleitband zum 22. VIP-Kongress", Rahman Jamal, Ronald Heinze (Hrsg.), VDE Verlag, ISBN 978-3-8007-4441-1
  - https://git.gsi.de/EE-LV/CSPP
- Successful Feasability Study: Mobile Agent based on LVOOP (LabVIEW 8.5)
  - https://wiki.gsi.de/foswiki/bin/view/NIUser/LVMobileAgentSystem
  - https://wiki.gsi.de/foswiki/pub/NIUser/LVMobileAgentSystem/DiplomarbeitFrederikBerck.pdf
  - https://forums.ni.com/t5/LabVIEW-Development-Best/Developing-a-Mobile-Agent-System-Using-LabVIEW-Object-Oriented/ta-p/3531213
  - https://forums.ni.com/t5/LabVIEW-Development-Best/Studying-the-Feasibility-of-an-Agent-System-Based-on-LabVIEW/ta-p/3531225

Lessons Learned – What Users say about *CS*

(from a talk by Stefan Götte, at the NI BIG PHYSICS Round Table, Paris, 2009)

- **Stress Field of the Responsible Person:**
- The OS (Win7) and/or the intranet are always unreliable,
- the programming language (LabVIEW) never really fulfills the need,
- the framework (*CS*) is only close to the requirement,
- the classes of other *CS* collaborators are typically not usable,
- there is no way to test things since there is no test system available while the real system is always in use,
- the users never define what the program has to do, but
- are not pleased with what the programmer delivers, and
- they misuse the system additionally.
- Anyhow: The goal is an easy system where the happy user does not realize what happens behind the scene, which works reliable and for ever (better: till the next LabVIEW version is installed).

Lessons Learned – The “Saving-Time-Myth”

- Wikipedia: “…software frameworks … reducing overall development time” (**?**)

- Amount of time for solving a problem
- decreases dramatically, if a problem may be solved with (generic) existing software. “configuration instead of coding”
- does not change – but the solution is much better
- may even increase (short-term), compared to a dedicated solution not (!) using the framework (required: training, courses, understanding and application of conventions)
- decreases (long-term): framework maintained by others, re-usability of code, replacing hardware, coding conventions enforced, maintainability, common language, know-how transfer, …