Generic and Re-usable Developments for Online Software

Slow Control, Configuration, Data Format & Online Processing

Shebli Anvar, CEA Irfu

January 12, 2011

CEA Irfu Saclay – Sédi/Lilas
Software Engineering Laboratory for Scientific Applications

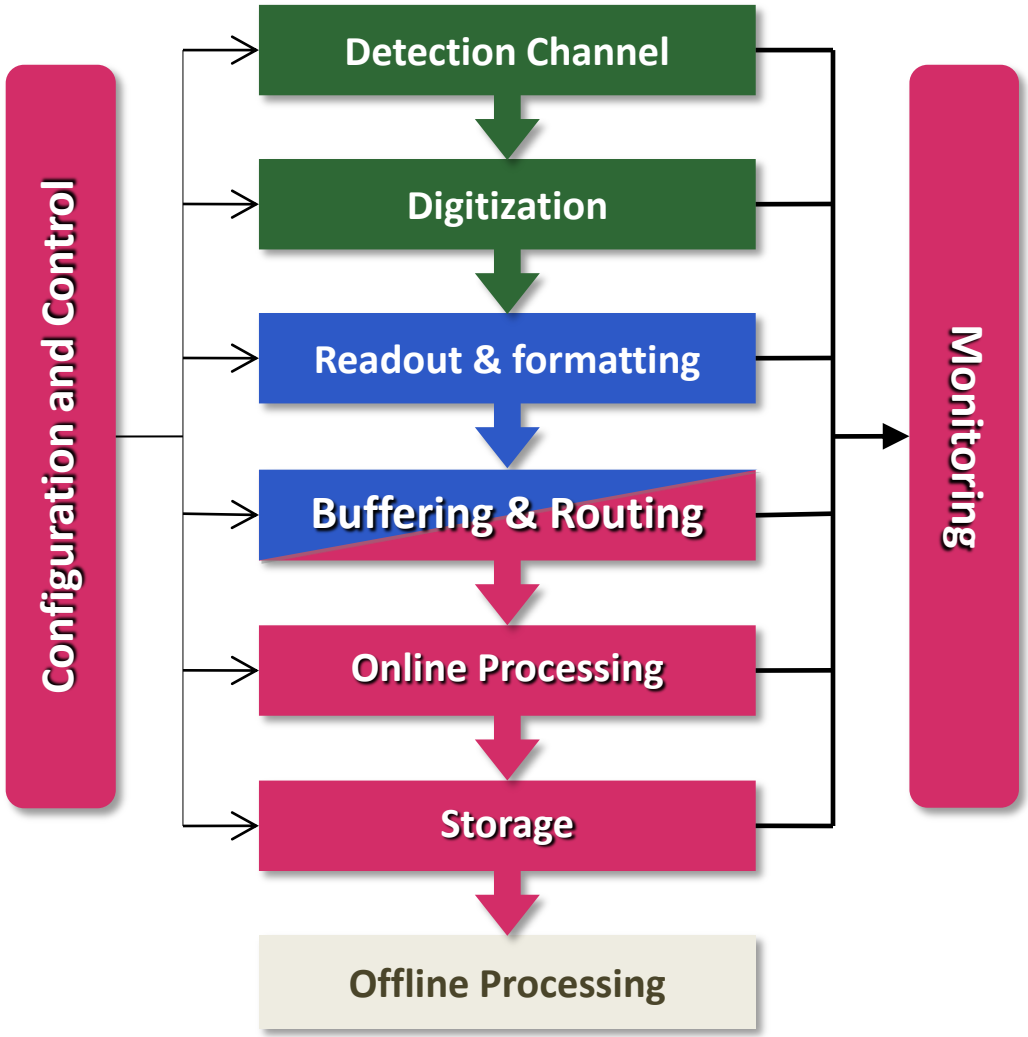
- **11 engineers + 1 technician**
- **All types of software development**
  - **Non Real Time**
    - HEP frameworks (LHC experiments, Antares, …)
    - Astrophysics frameworks (Herschel, Svom, …)
    - Physics simulations (cosmology, plasma physics, HEP, …)
    - Detector simulations (Geant4, …)
    - …
  - **Real Time / Online**
    - Embedded / FPGA driver software
    - Configuration, Control & Monitoring
    - Data acquisition and online processing
    - GUIs
    - …
  - **Other software related activities**
    - Development tools (Subversion, IDE, …)
    - Web portal management
    - …

4 engineers
3 on generic frameworks

Real-time or Online Software Developments

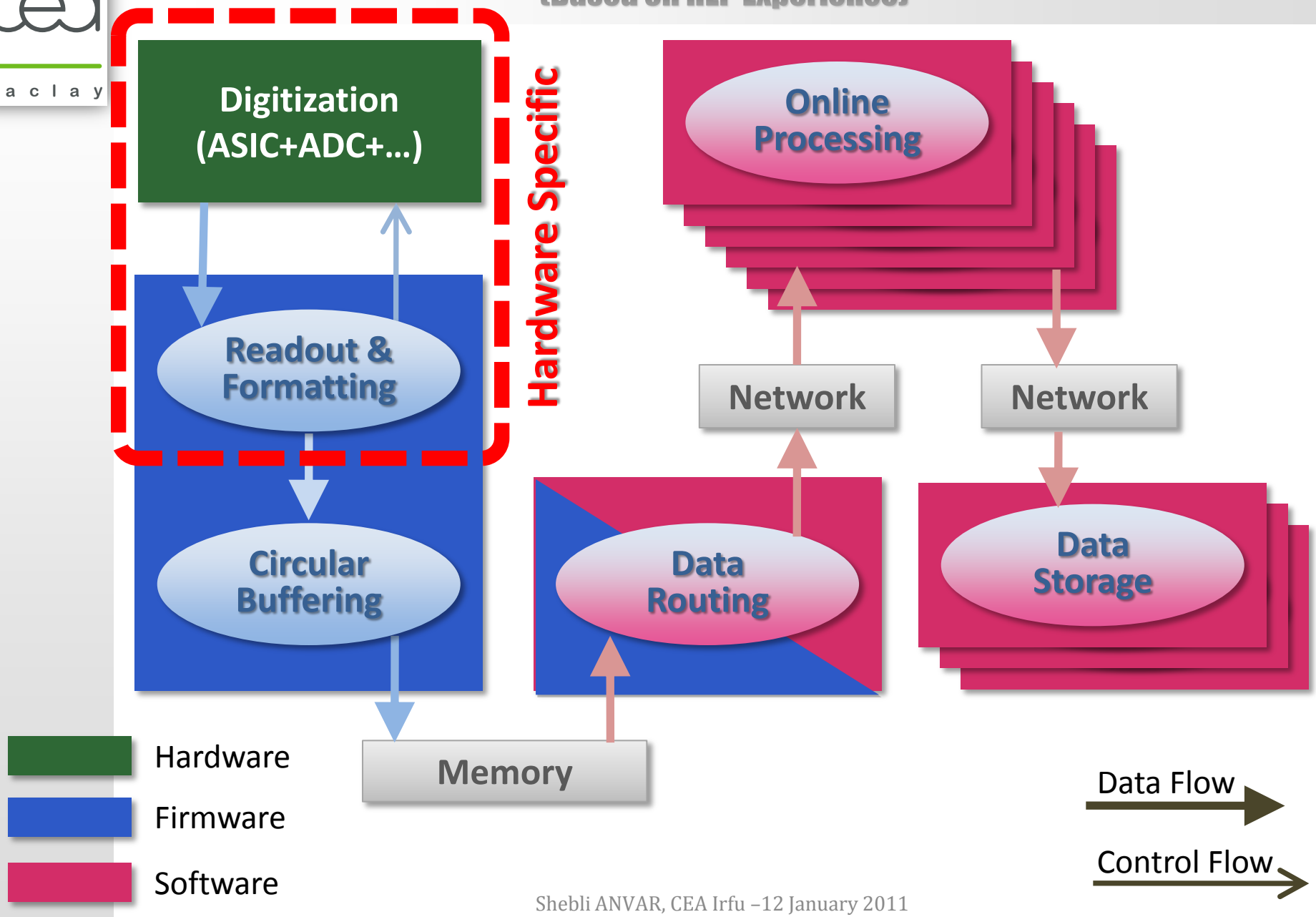
- Antares / KM3NeT (underwater neutrino telescope)
  - SCOTT ASIC test bench
  - Offshore data acquisition system
  - Slow control and configuration
- Svom (Gamma ray burst satellite and telescopes)
  - Ground segment GRB alert evaluation and dispatch
  - Eclair gamma detector test bench (TRAPS lab)
- T2K Test bench + many other projects using AFTER electronics in particle physics, astrophysics and nuclear physics
- Forfire (outdoors fire detection)
- GET
- Possibly: CLAS12, S3, GBAR, …

Control, Acquisition & Online Processing Chain

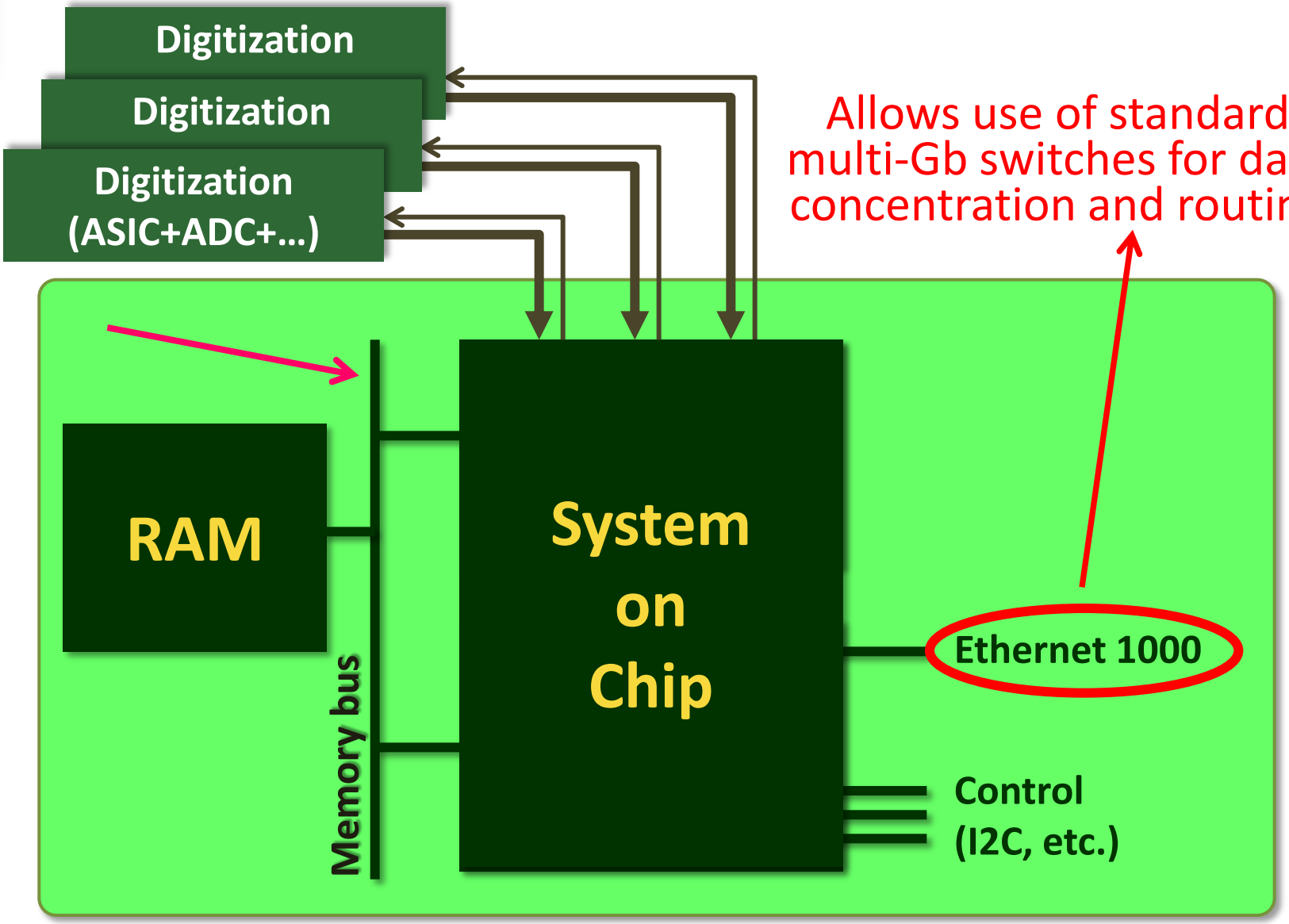


Typical Acquisition Architecture

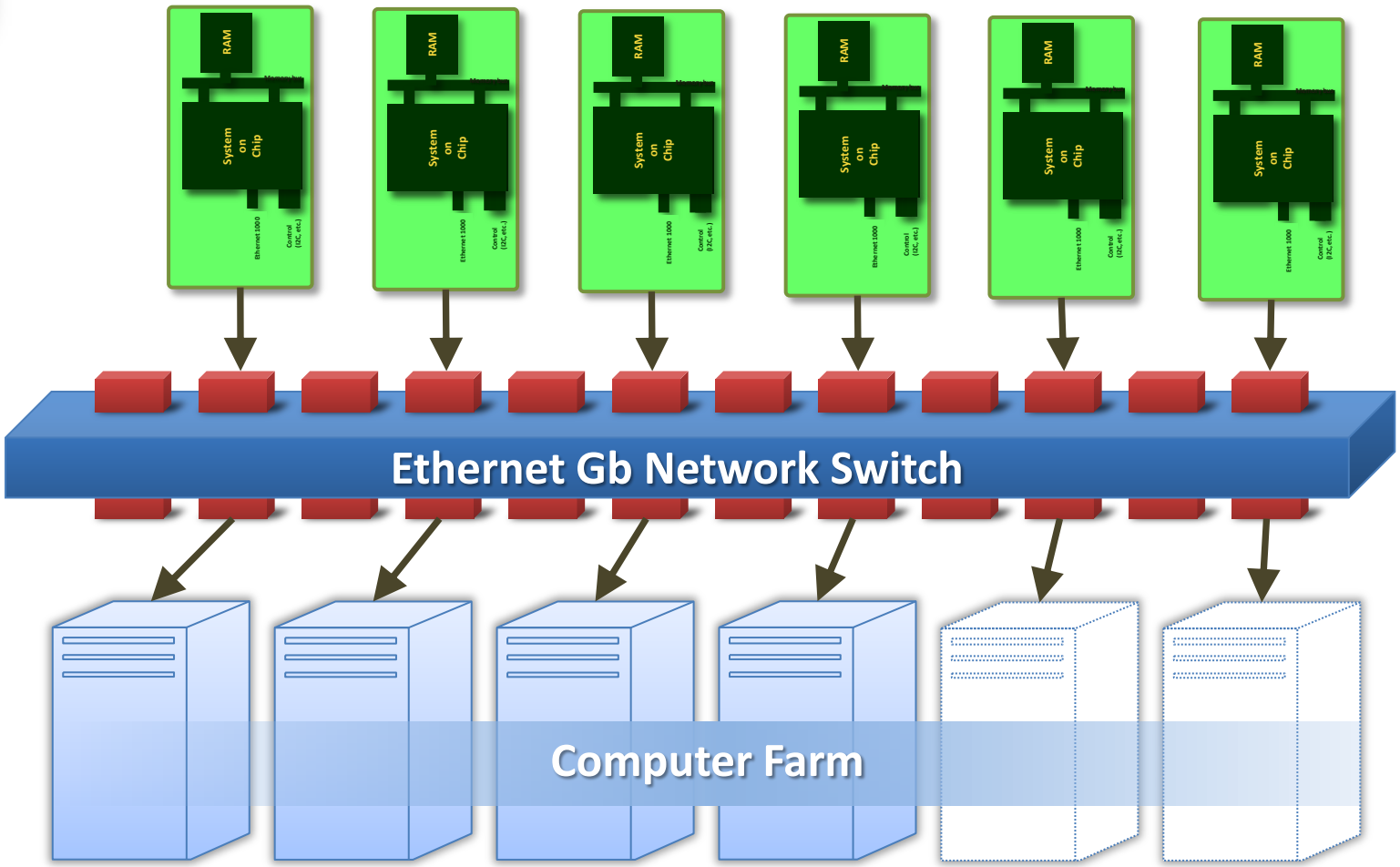
(Based on HEP Experience)



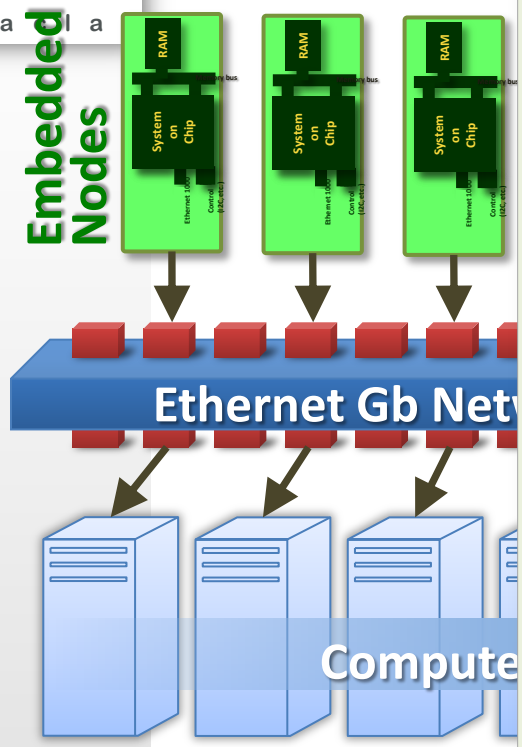
Evolution of Embedded Implementation (Upstream)

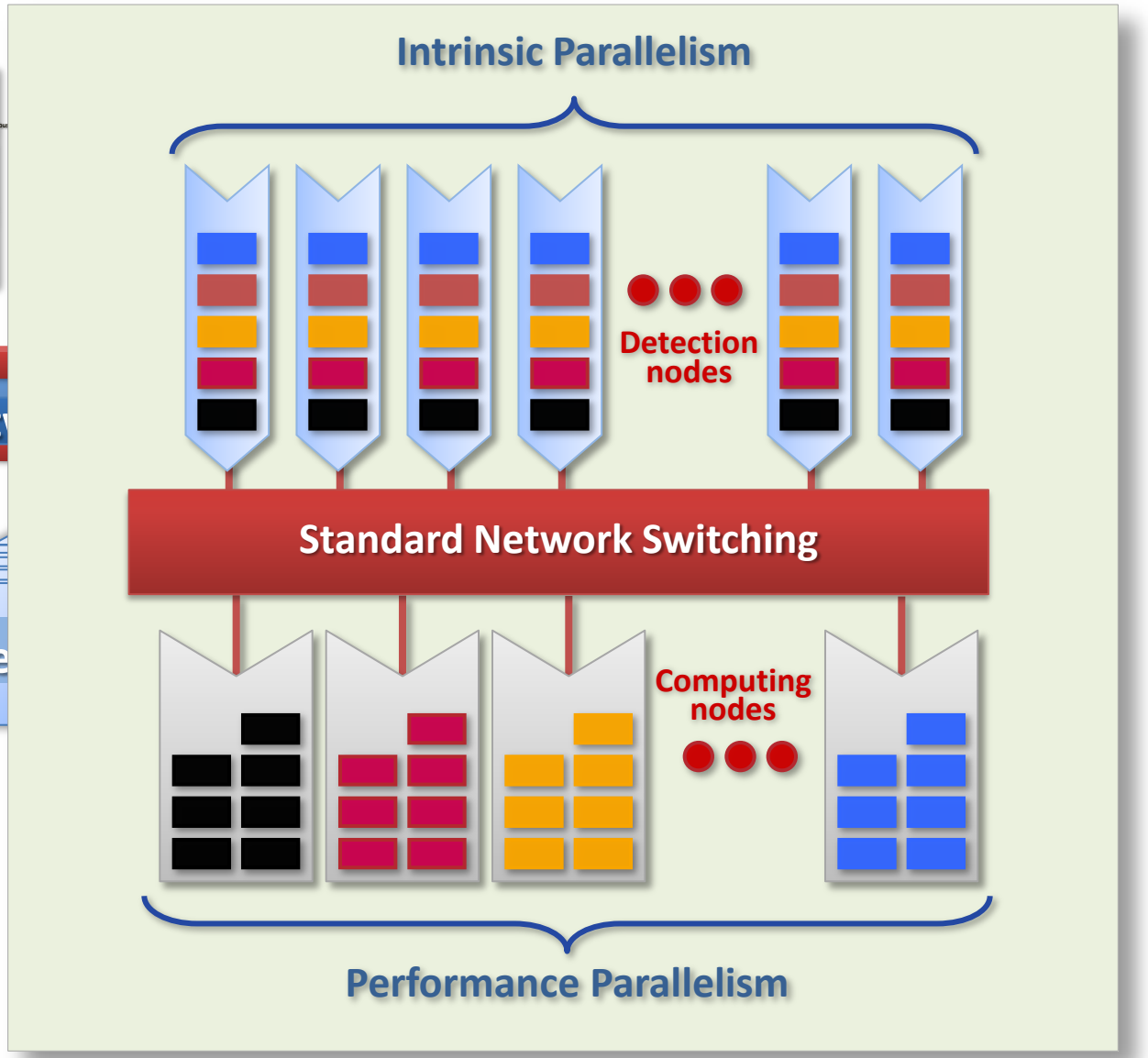


Data Flow Infrastructure

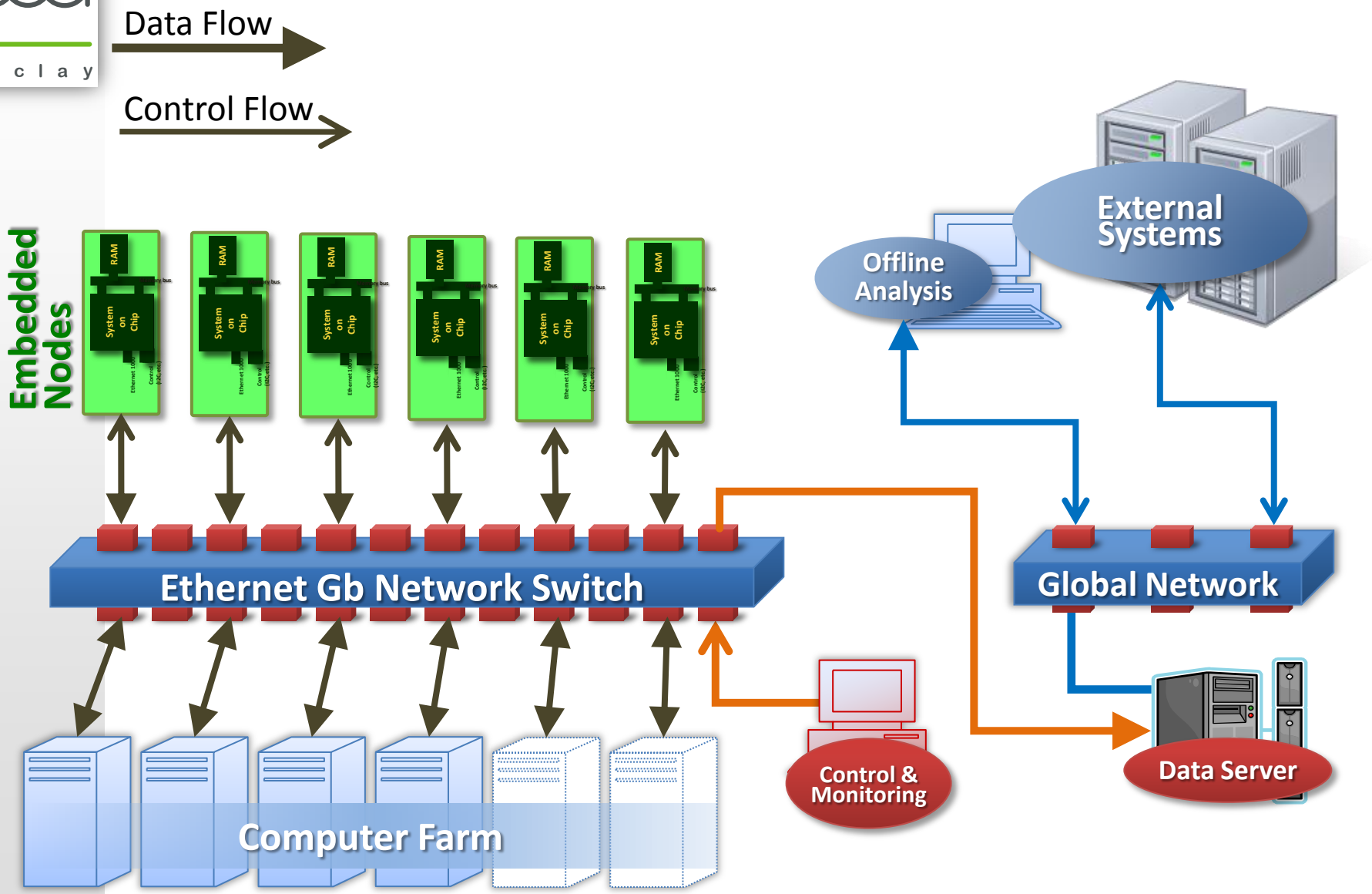


Time-slice Building Paradigm





Global View of DAQ Hardware Infrastructure



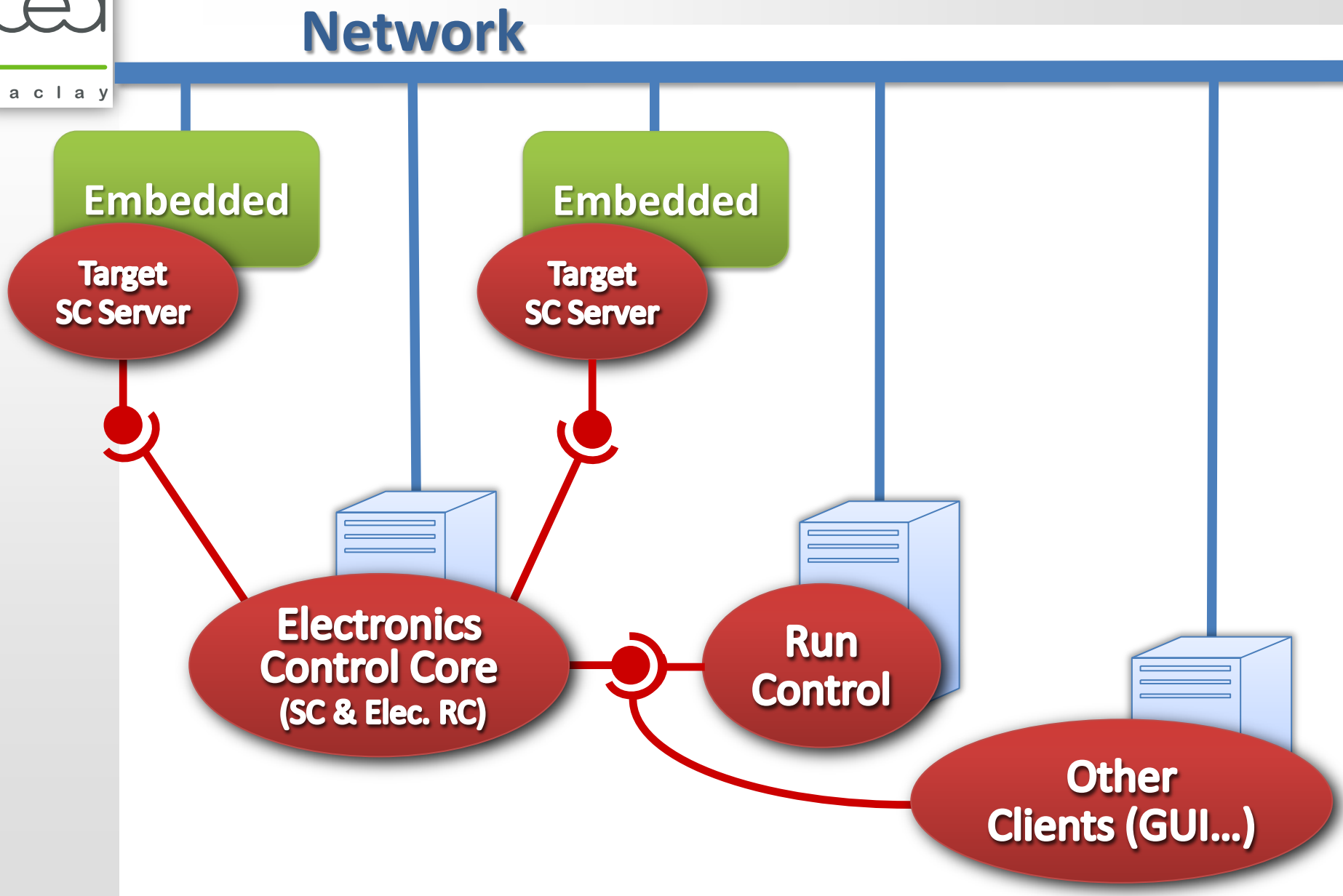
Software Engineering for Distributed Development

- **Provide for easy software integration**
  - Tools for integration of subsystems configurations
  - Use Client-Server architecture supported by multi-language middleware
  - Define interfaces using a formal language (WSTL, Slice, Corba IDL…)
- **Accomodate multiple test benches / integration sites**
  - Automatically share common information through central DB (e.g. calibration values)
  - Provide for easy DB access

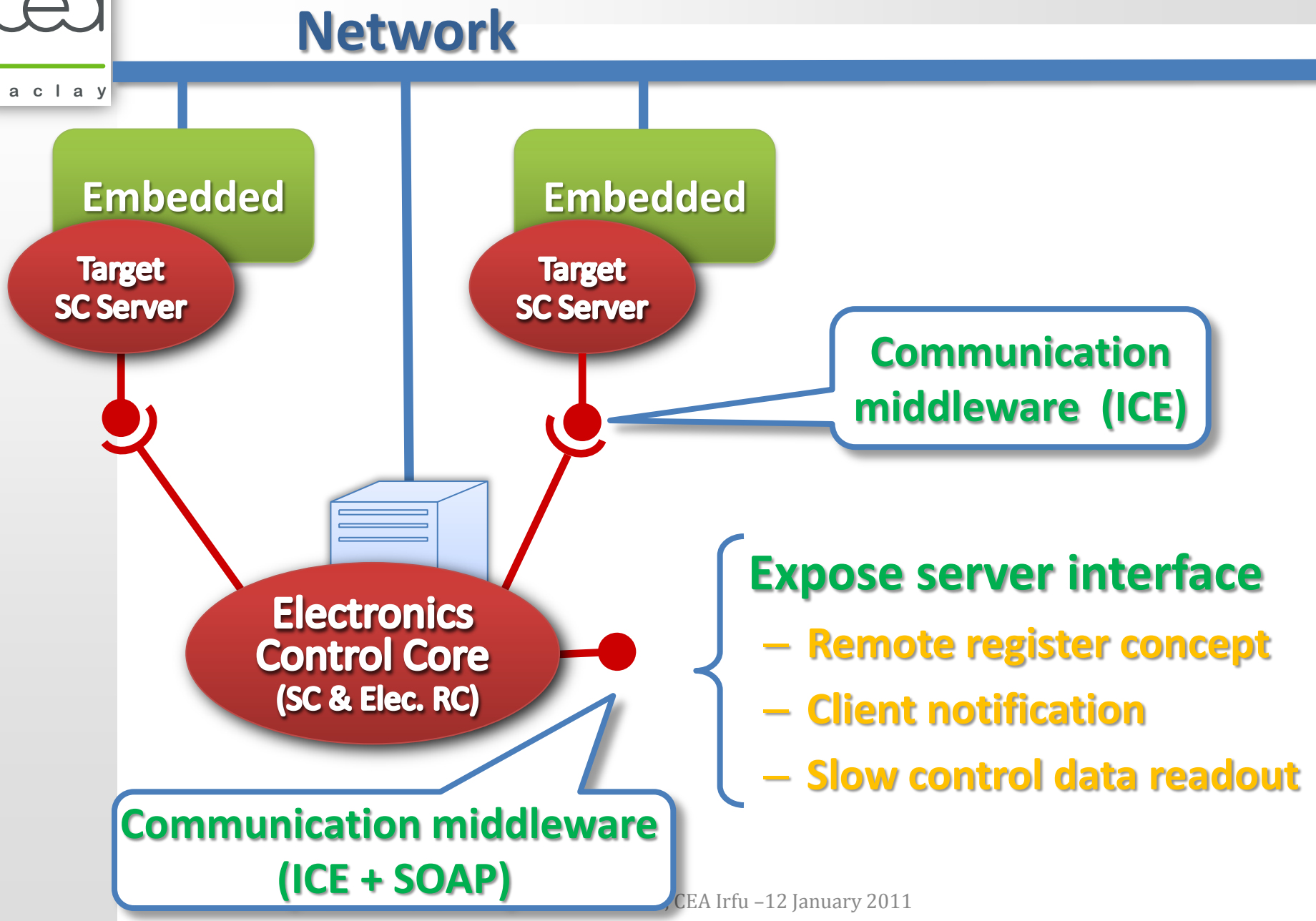
Software Technologies for Distributed Applications

- Middleware for distributed multi-language control and configuration (ZeroC Ice, SOAP/ Web service, …)
- Middleware / framework for distributed fast data acquisition (Bare TCP/IP, Optimized ZeroC Ice, …)
- Remote control, configuration and monitoring (Google Web Toolkit, Qt+ZeroC Ice, …)
- Distributed access to configuration and run conditions through database.

Electronics Control Core Concept



Electronics Control Core Concept



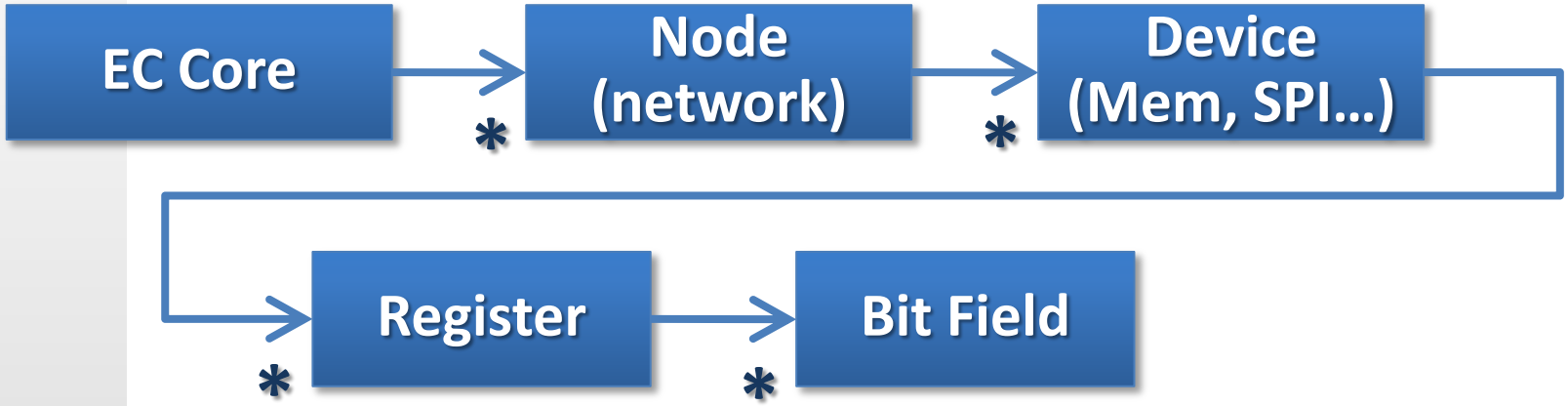
Expose server interface

- Remote register concept
- Client notification
- Slow control data readout

Electronics Control Core Concept

Expose server interface

- Remote register concept
- Client notification
- Slow control data readout



ICE Interface & data definitions

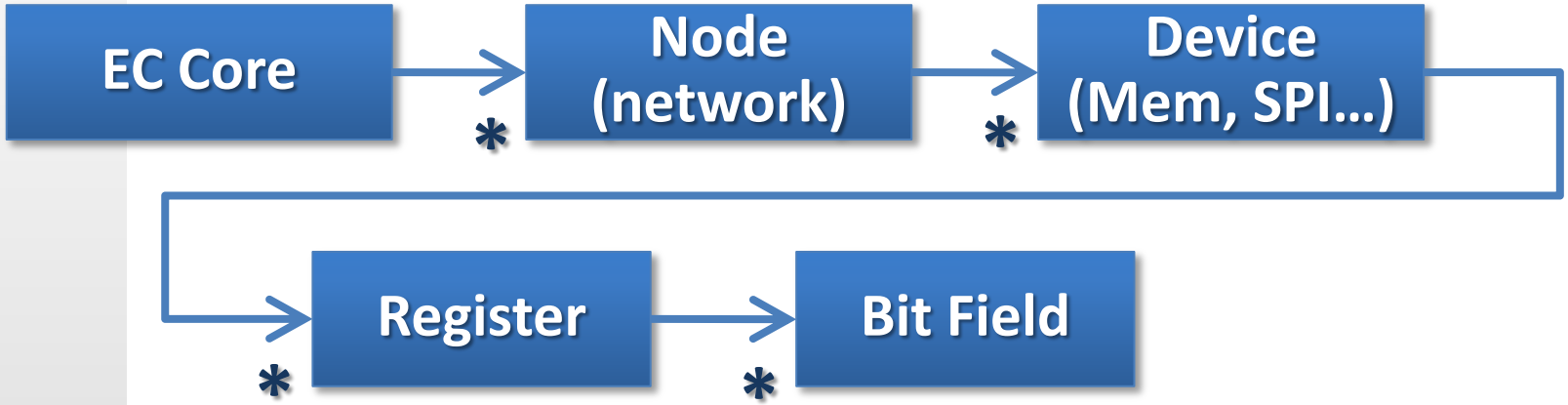
Embedded (VxWorks) C++ library

Host (Linux) C++ library

Electronics Control Core Concept

Expose server interface

- Remote register concept
- Client notification
- Slow control data readout

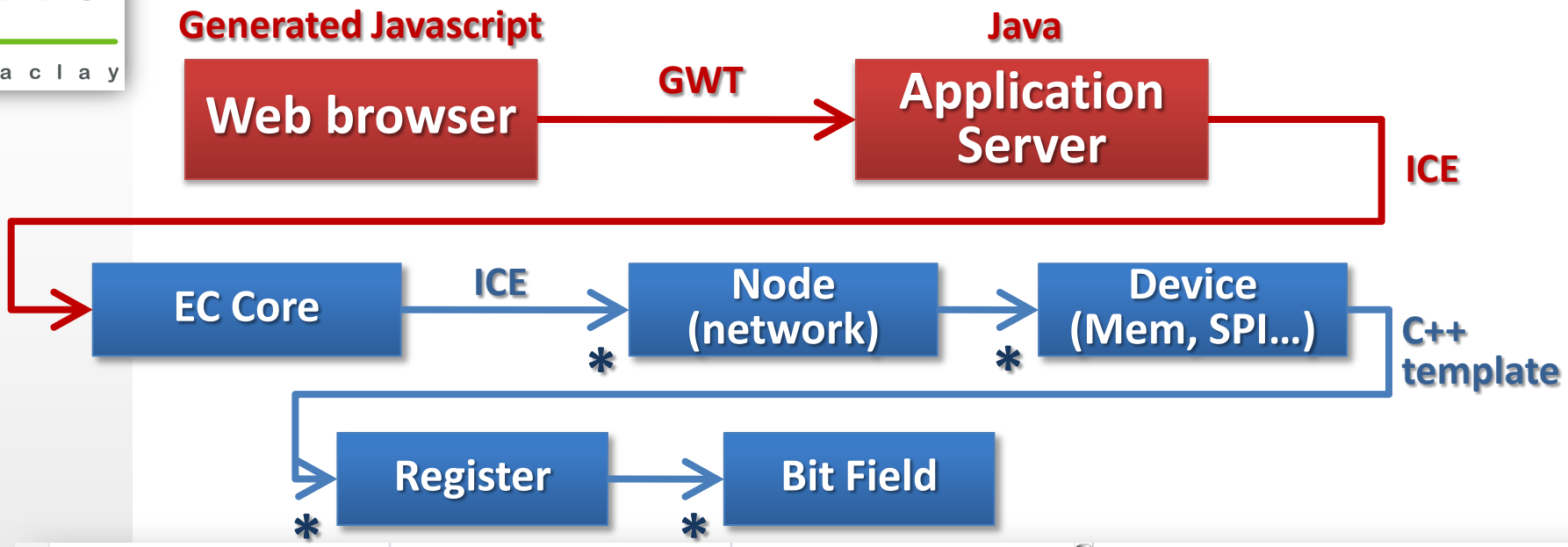


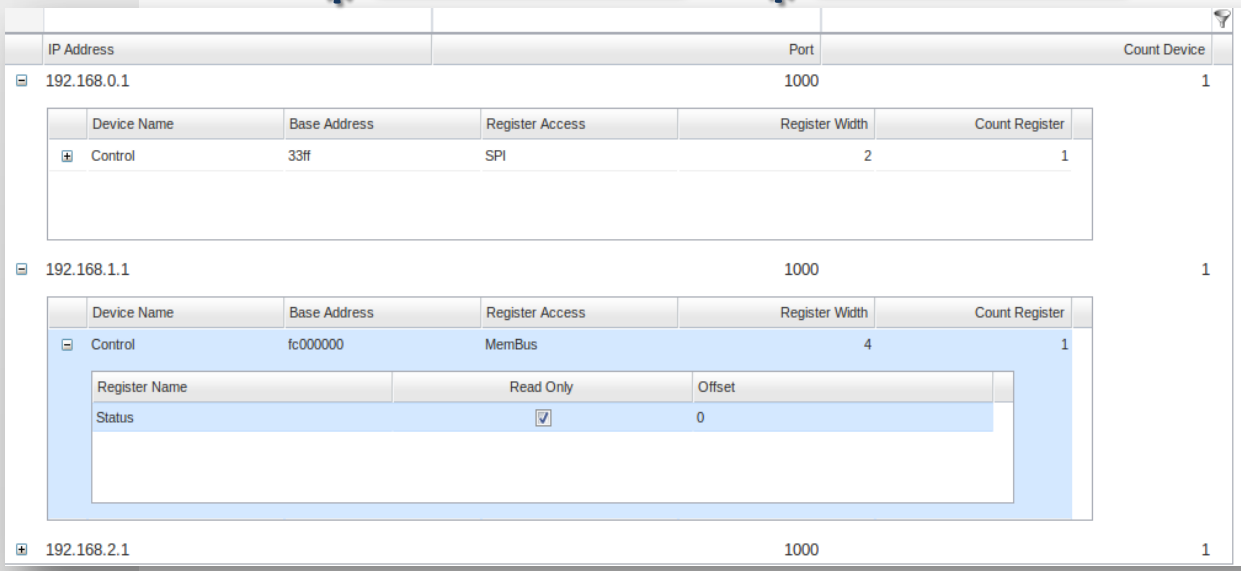
Use of C++ templates for

- register data type (byte width)
- register access policy

memory map, SPI, simulator, etc.

Electronics Control Core Web Access





IP Address	Port	Count Device
192.168.0.1	1000	1

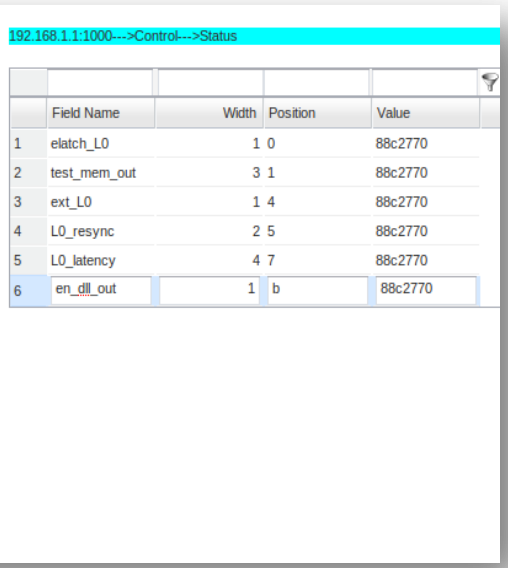
Device Name	Base Address	Register Access	Register Width	Count Register
Control	33ff	SPI	2	1

IP Address	Port	Count Device
192.168.1.1	1000	1

Device Name	Base Address	Register Access	Register Width	Count Register
Control	fc000000	MemBus	4	1

Register Name	Read Only	Offset
Status	☑	0

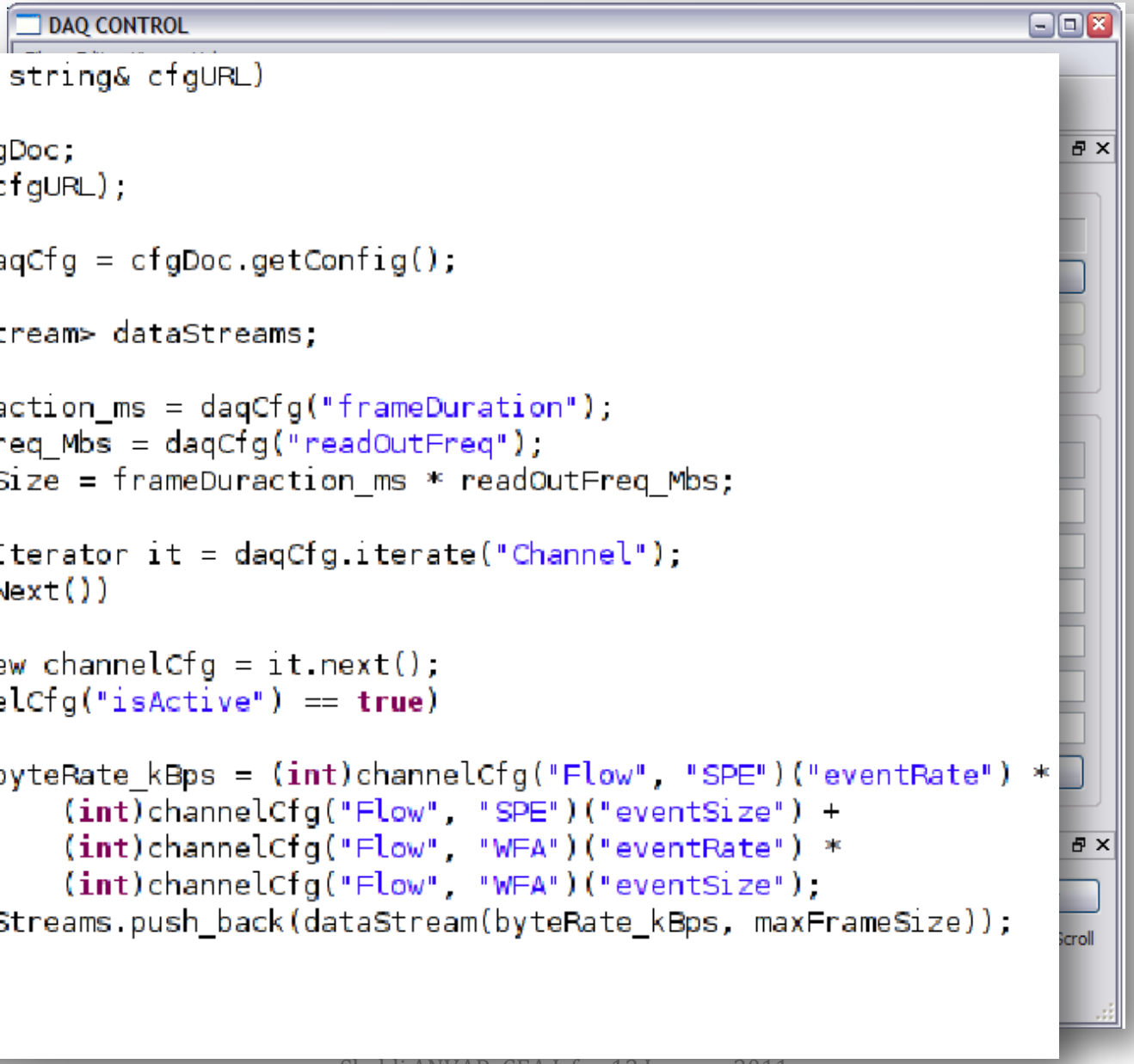
IP Address	Port	Count Device
192.168.2.1	1000	1



192.168.1.1:1000--->Control--->Status

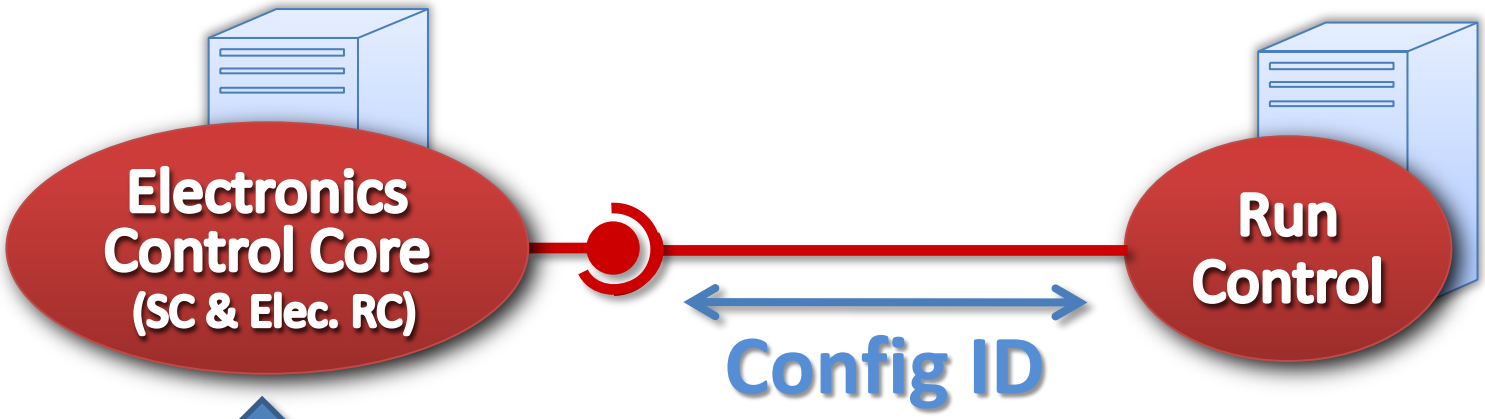
	Field Name	Width	Position	Value
1	elatch_L0	1	0	88c2770
2	test_mem_out	3	1	88c2770
3	ext_L0	1	4	88c2770
4	L0_resync	2	5	88c2770
5	L0_latency	4	7	88c2770
6	en_dll_out	1	b	88c2770

CompoundConfig Configuration Framework




```
initialize(const string& cfgURL)
{
    CCfg::Doc cfgDoc;
    cfgDoc.load(cfgURL);

    CCfg::View daqCfg = cfgDoc.getConfig();

    vector<DataStream> dataStreams;

    int frameDuraction_ms = daqCfg("frameDuration");
    int readOutFreq_Mbs = daqCfg("readOutFreq");
    int maxFrameSize = frameDuraction_ms * readOutFreq_Mbs;

    CCfg::View::Iterator it = daqCfg.iterate("Channel");
    while(it.hasNext())
    {
        CCfg::View channelCfg = it.next();
        if(channelCfg("isActive") == true)
        {
            int byteRate_kBps = (int)channelCfg("Flow", "SPE")("eventRate") *
                    (int)channelCfg("Flow", "SPE")("eventSize") +
                    (int)channelCfg("Flow", "WFA")("eventRate") *
                    (int)channelCfg("Flow", "WFA")("eventSize");
            dataStreams.push_back(dataStream(byteRate_kBps, maxFrameSize));
        }
    }
}
```

Configuration Logic

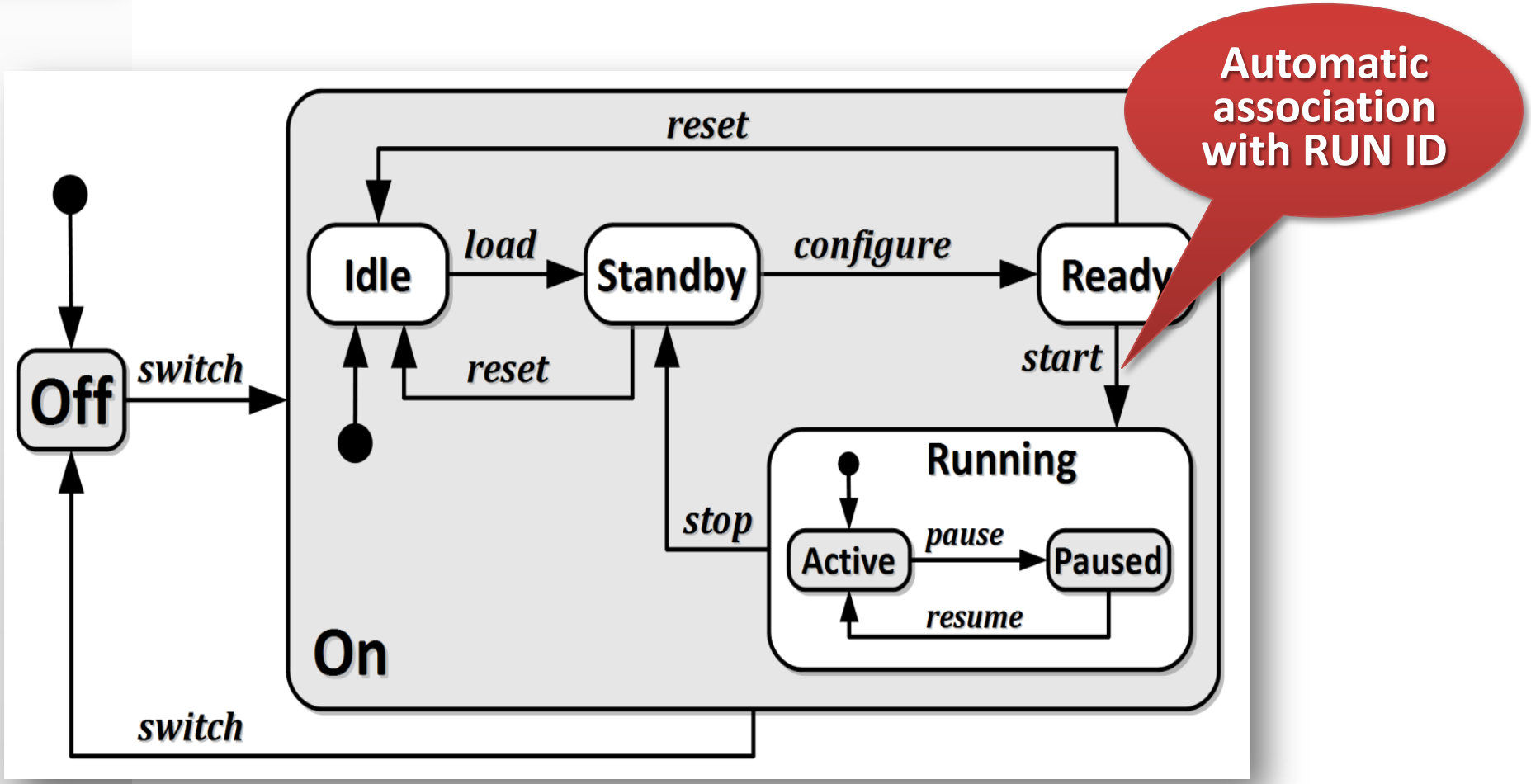


Config
save/restore

Configurations

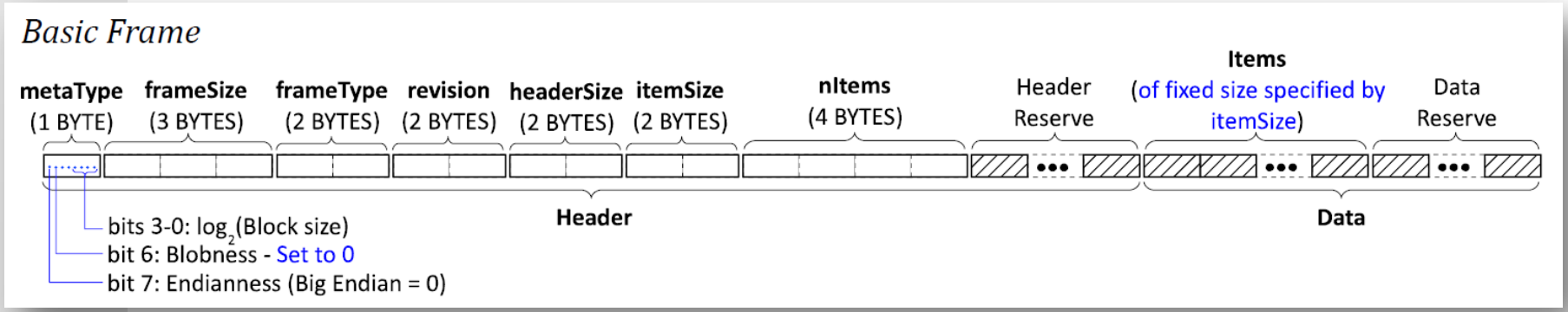
- C++ Framework (Host & embedded)
- Multilanguage server access (using ICE)
- Java wrappers
- Automatic Config Tree → DB mapping
- Automatic association with RUN ID

Configuration Logic



DAQ Data Format

- **Multiframe Metaformat**
  - Generic, versionable DAQ data format
  - Provides for format evolution
  - C++ implementation
  - Java implementation?
  - Description “language” + code generation?



Focus on engineering strategy & methodology!

- **Think reuse**
  - Use open source frameworks
  - Design your own software as a framework and patterns
  - Professionalize your code (versioning, doc, build system, etc.)
  - Provide for (foresee) multiple platforms
  - Try pattern reuse with software interfaced firmware
- **Think flexible**
  - Design functionalities as services to clients
  - Base your design on OO components
  - Provide for multiple languages (esp. middleware)
  - Integrate many simple frameworks (one for every aspect) rather than relying on a big one

Ongoing Online Framework Developem
(for embedded targets and worksta

- CompounConfig C++ / Java Framewor Doc: 80%)
  - Flexible advanced configuration tool
  - Both file and DB mappings for pers
  - Graphical configuration editing
  - Run conditions database
- MDaq Framework (70% %)
  - Distributed electronic emote register access)
  - Dynamic develop n configuration (no recompilation)
  - Multilanguage Java, Python, etc.)
  - Multiframe (backward & forward compatible data format)
- KDaq Fra 0% - Doc: 10%)
  - Re-u le DAQ processes (Data routing, Farm processing, Sc ge, Run control GUI, Web GUI)

Today's distribution mode: ask me for a tarball (s.anvar@cea.fr)
Licencing: GPL-like Cecill (special contract for proprietary licencing)