Abeans Application Framework and Widgets Running with EPICS

M.Plesko on behalf of **KGB Team**
J. Stefan Institute and Cosylab Ltd.
in collaboration with **SNS** and **DESY**

cosylab
CONTROL SYSTEM LABORATORY

EPICS Workshop,
Berlin, May 2002

eManagement Cycle

cosylab
CONTROL SYSTEM LABORATORY
www.cosylab.com

Project definition

Internal Pages

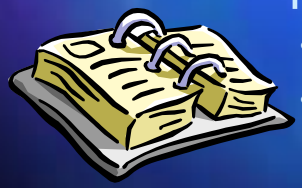
- •open project
- •select project manager
- •select people included

CS User

Project Web

- •Sources
- •Manuals
- •...

Scripts

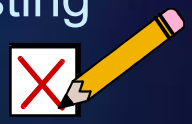


Request Tracker

- •To-Do
- •Activity list

Bug report

Testing



Actual work

- •Code
- •XML Docs

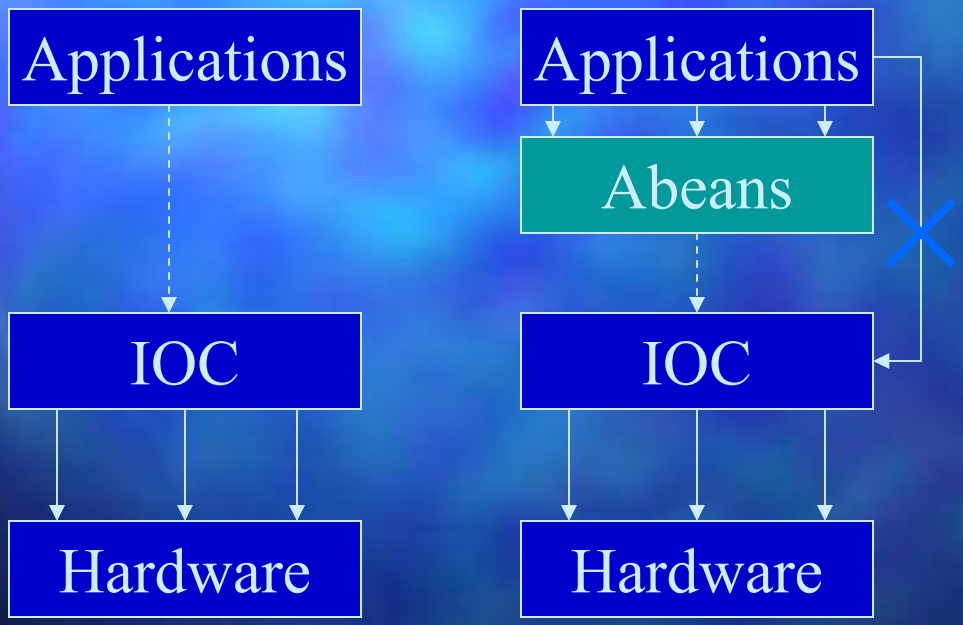
Source Repository

- • Sources
- • Manuals (in XML)
- • Resources

Abeans (or xal at SNS)

cosylab
CONTROL SYSTEM LABORATORY
www.cosylab.com

- Application framework

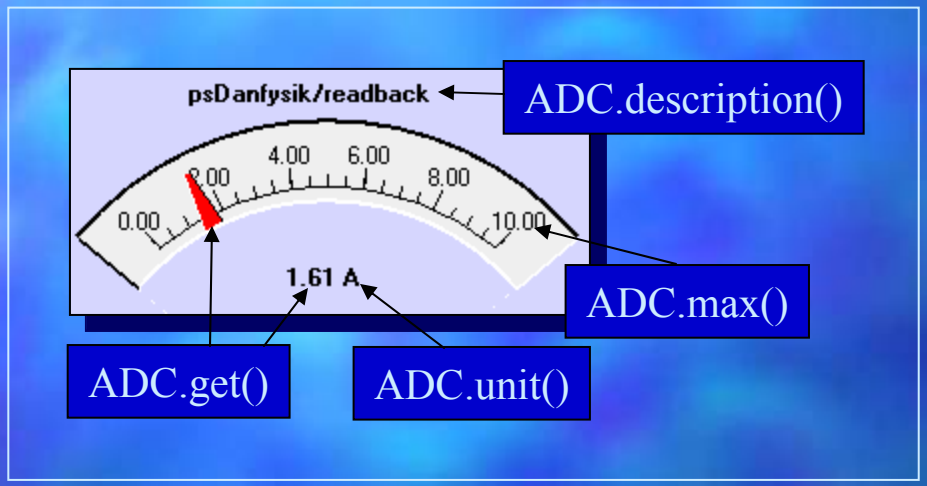


- Hide comm layer
- Provide application services
- Provide CS models (device, channel, etc.)

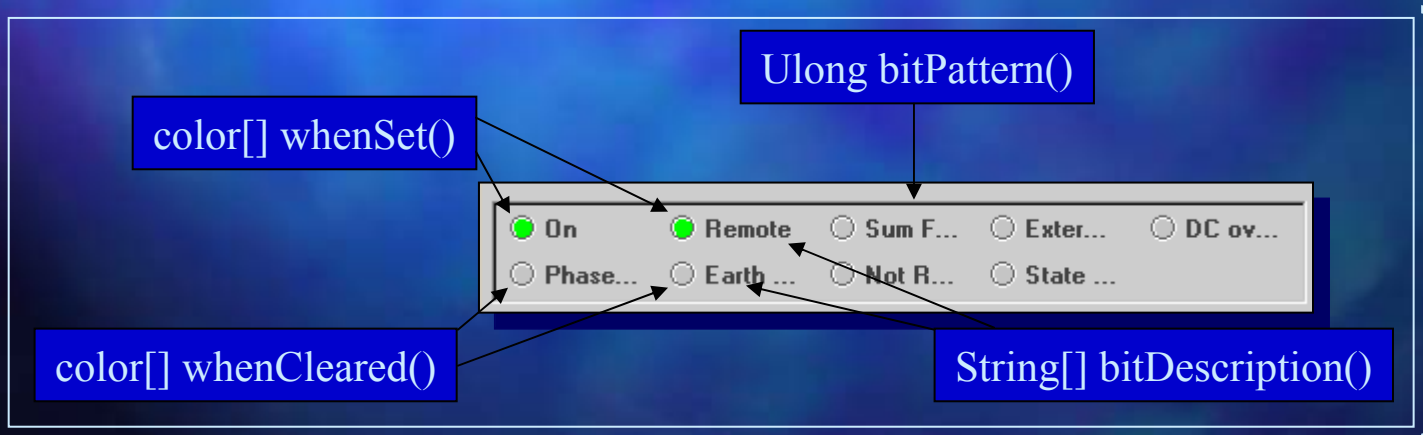
Σ: RAD, easier maintenance

CosyBeans Widgets

cosylab
CONTROL SYSTEM LABORATORY
www.cosylab.com

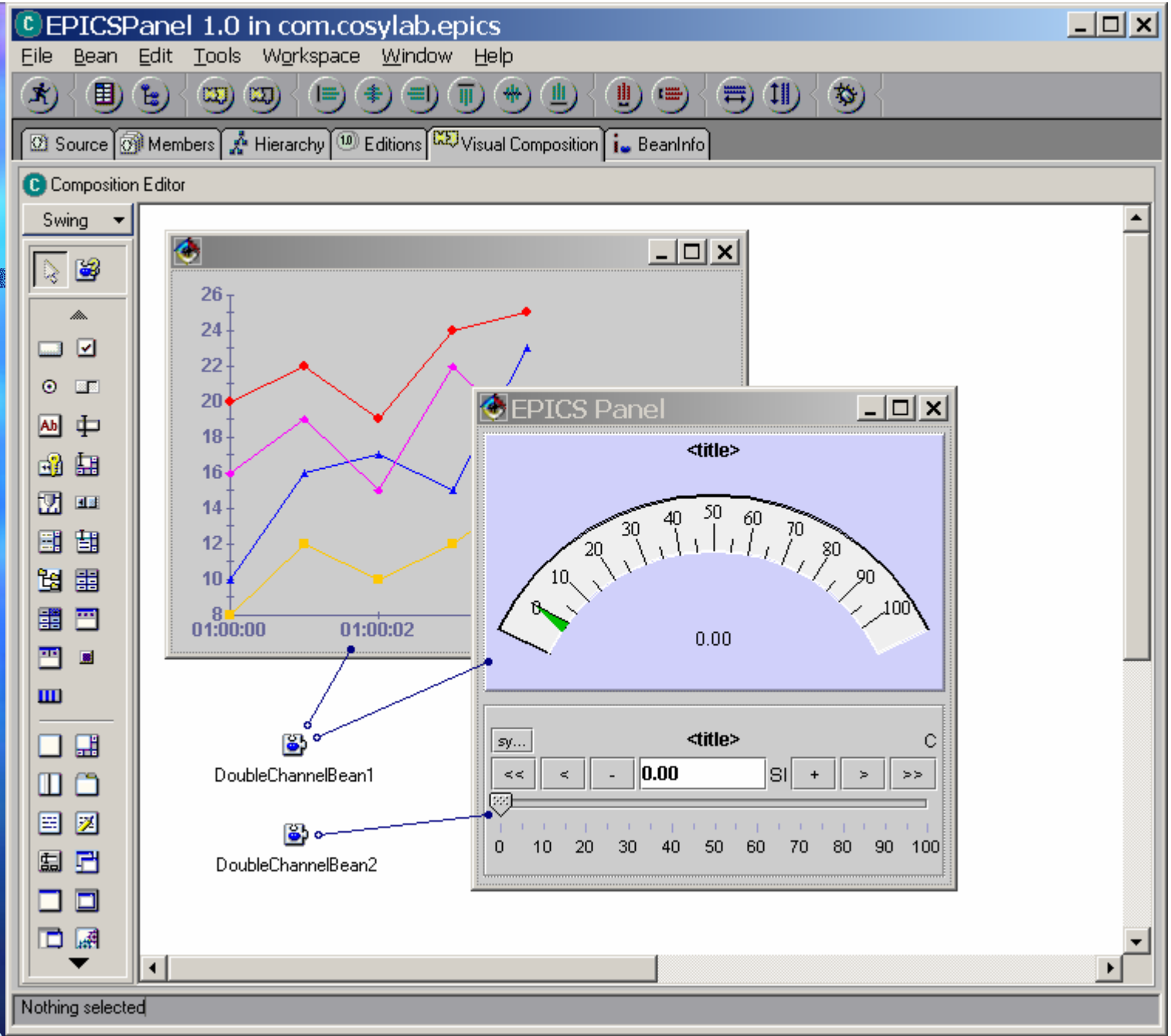


Gauge

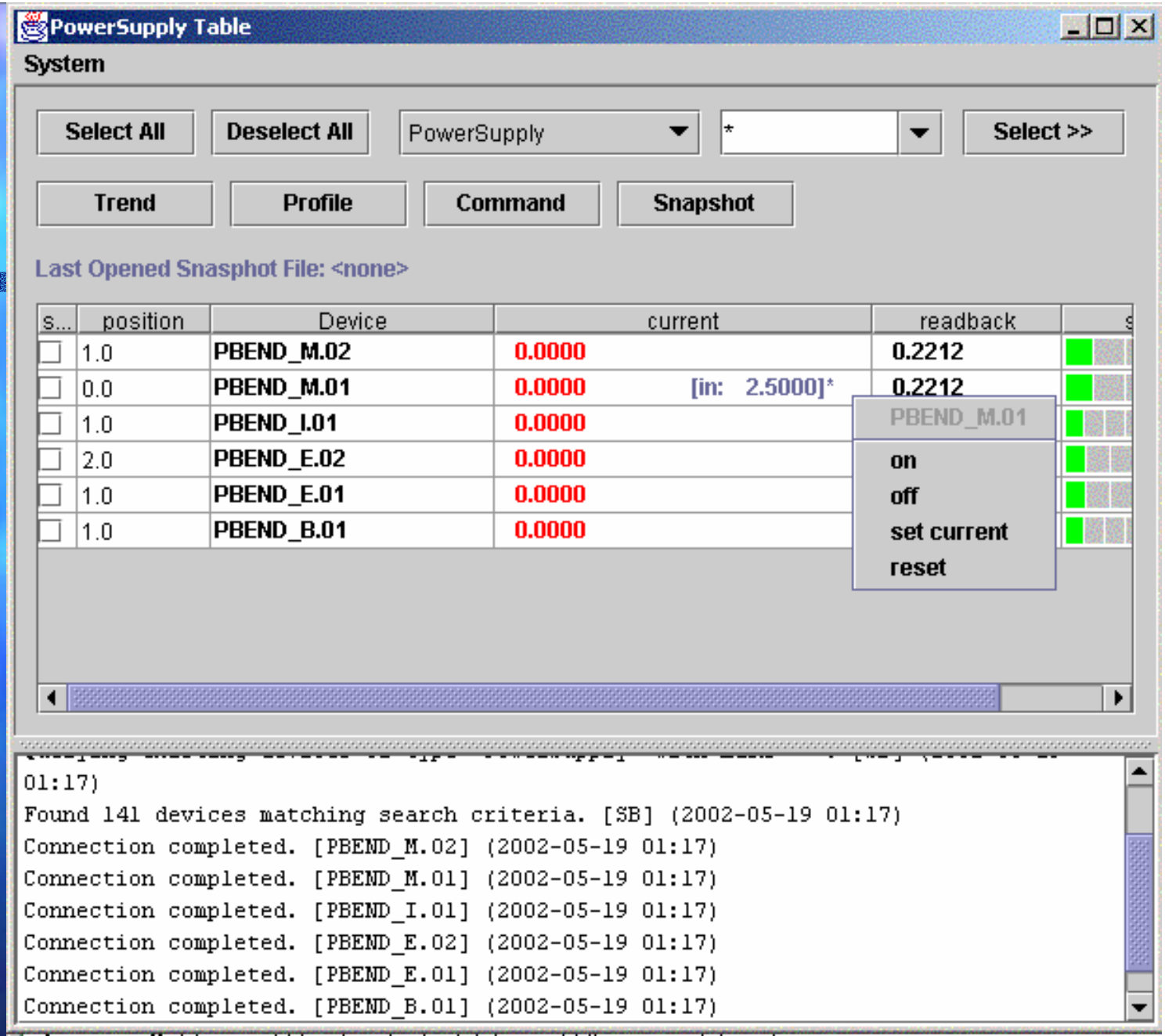


Status

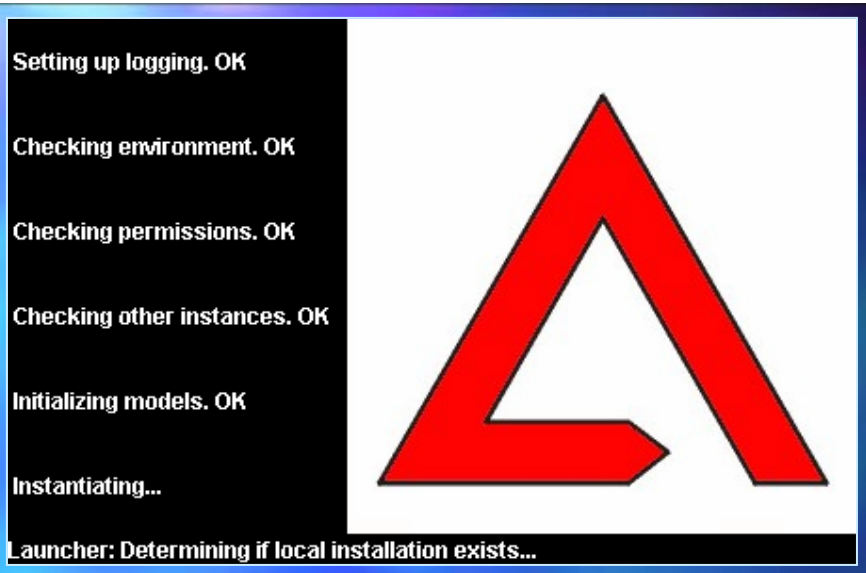
Visual Composition



Device Table



Launcher



Abeans +

CosyBeans

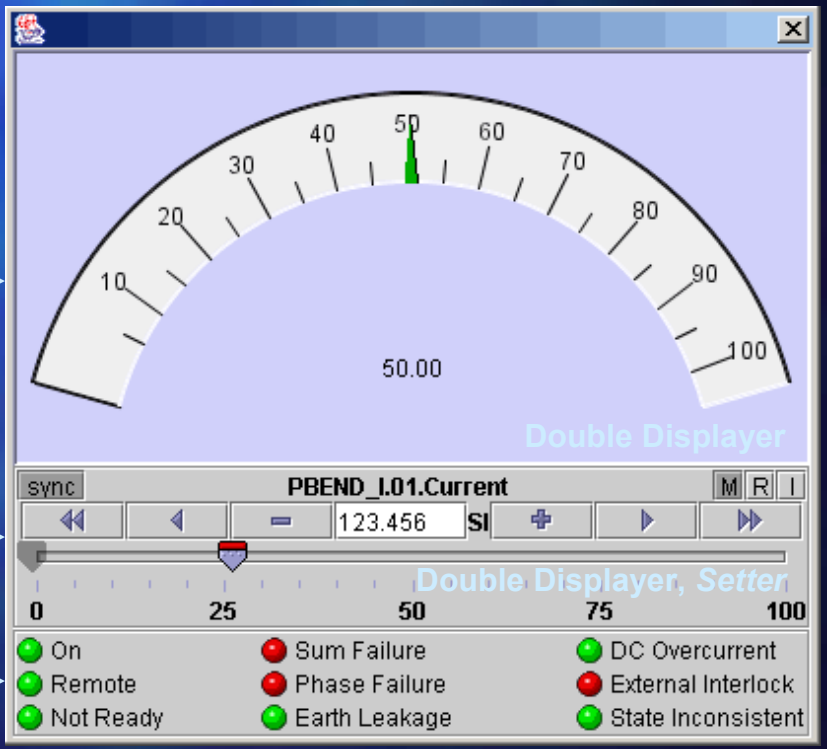
EPICS Plug (JCA)

Abeans

Double Adapter

Double Adapter *Setter*

Pattern Adapter



CosyBeans Features

- Optimized for Control Systems, not just GUIs
  - A result of long thinking, designing, comparing, testing, use at other laboratories, reconsiderations, eg.
    - initialize from CS at run-time
    - control system connection status+timeout handling
    - colour coding of states and alarms
  - Consistent behaviour of widgets in all panels
    - no subtle differences to learn
    - user can fully configure at run-time
    - resize-able, adjust shape to fully use available space
    - intuitive/safe modification of CS parameters (clever slider,...)
    - trend everywhere: history, save as CSV, histogram, zoom, ...
  - No duplication of programming efforts

Abeans Features

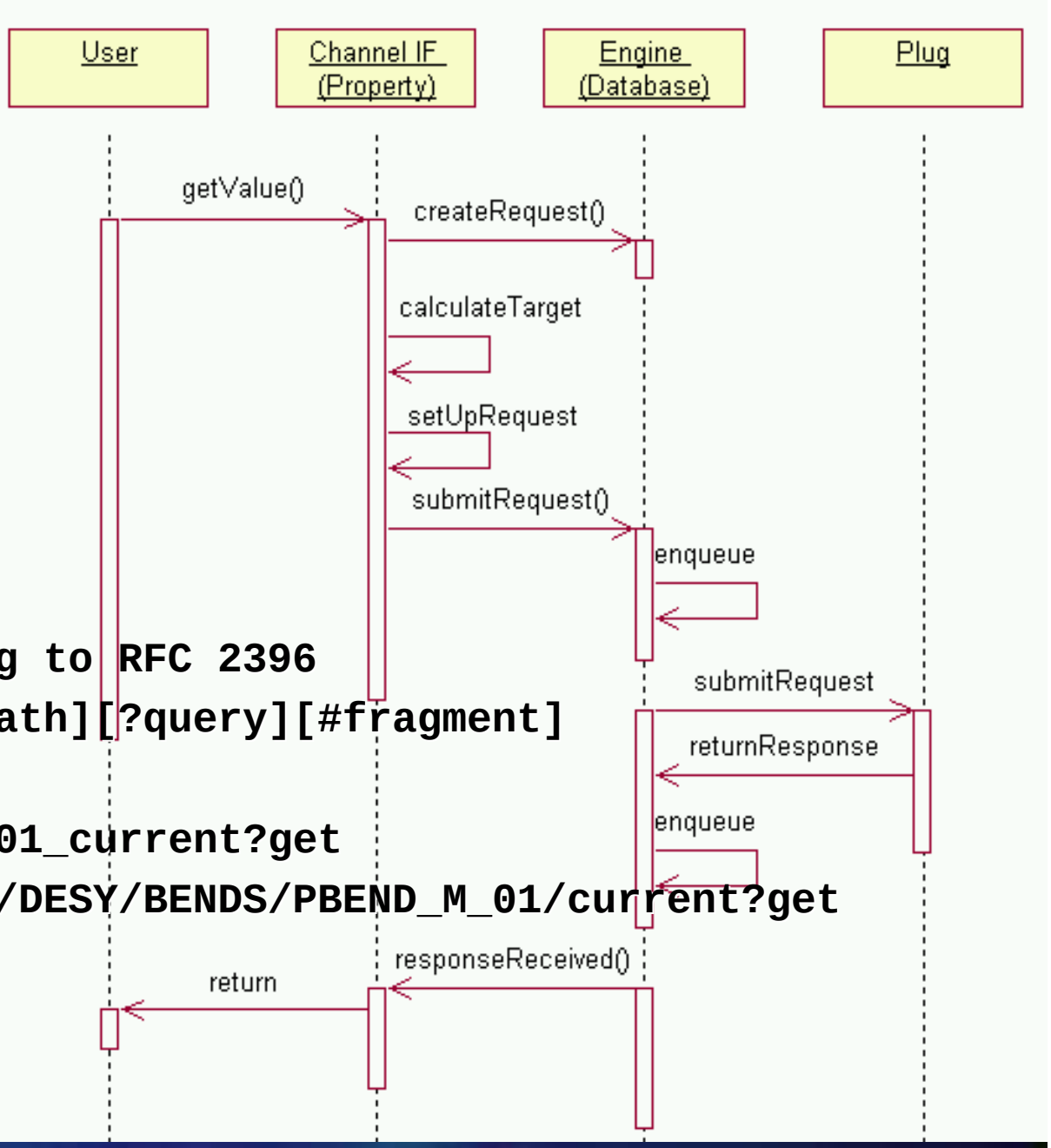
- Hide details of Control System - just one of many:
  - launch application/applet, remote install
  - Logging, alarm, and communication error reporting
  - Resource initialization and destruction, provide defaults
  - find, choose and connect to selected device
  - supports different protocol via plugs (e.g. CORBA, EPICS,…)
- Same core code is used everywhere
  - The code is tested very well
- Visual composition and normal programming together
  - choose device type and command through menu!
  - almost all errors detected by compiler not during operation
  - Abeans discover vis/man mode => change behaviour

Abeans and EPICS ?

cosylab
CONTROL SYSTEM LABORATORY
www.cosylab.com

- MEDM is faster for "expert" panels, TCL is simpler for prototyping, but:
- Abeans application development is scalable!
  - profit from standard Java features
    - GUI layout, file I/O, XML, WebStart, resources, logging,…
  - Abeans fight Java deficiencies
    - optimize graphic performance and reduce memory usage
- Use Abeans for complex applications…
  - client-side algorithms (machine physics)
  - display of mutliple data (tables, tree, bird-view)
- …and when long-term maintenance matters

ca_get()



use URI mapping according to RFC 2396

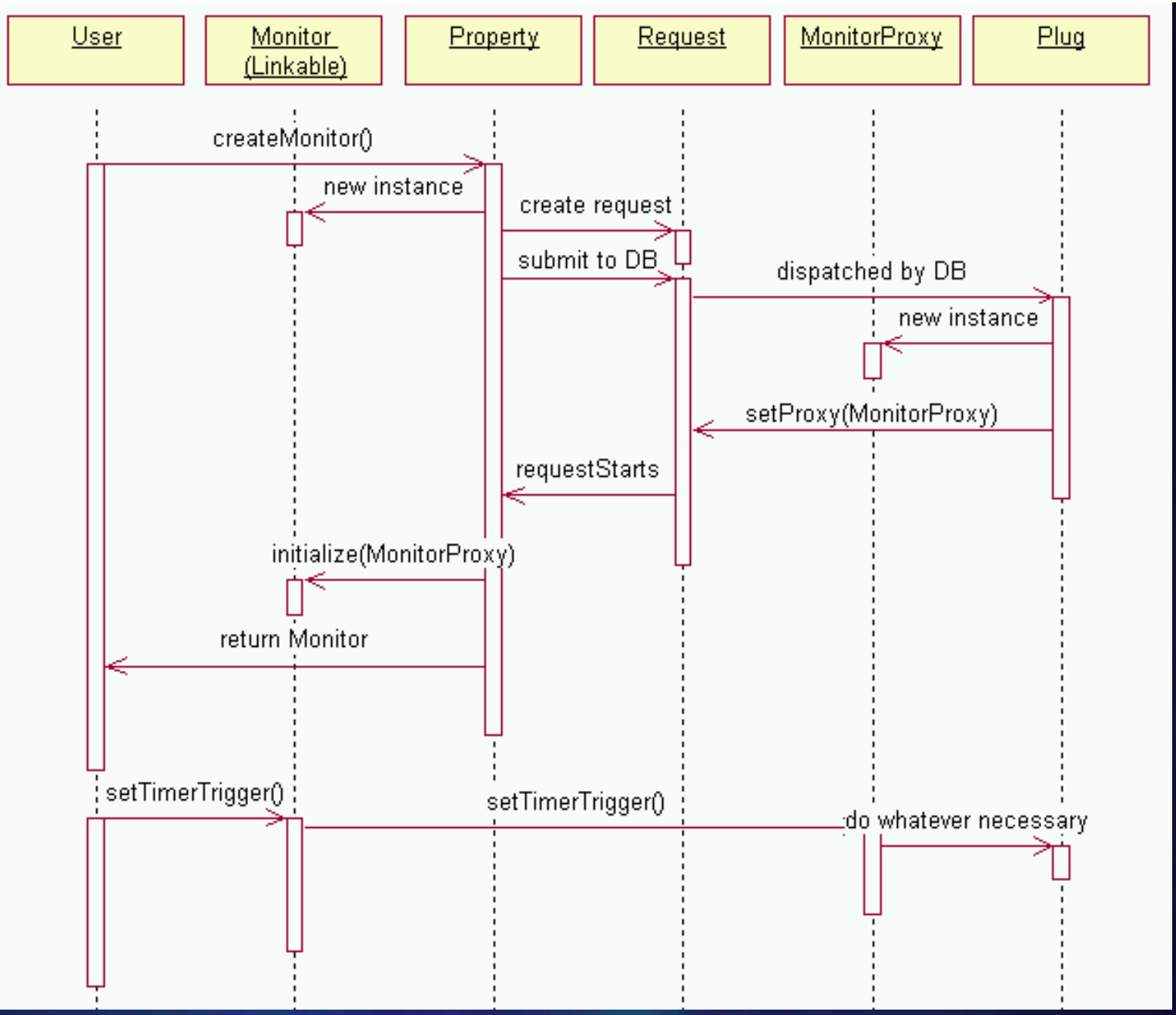
[scheme:][//authority][path][?query][#fragment]

e.g.

abeans-EPICS:///PBEND_M_01_current?get

abeans-TINE://ns.desy.de/DESY/BENDS/PBEND_M_01/current?get

Monitor


User
Monitor (Linkable)
Property
Request
MonitorProxy
Plug
createMonitor()
new instance
create request
submit to DB
dispatched by DB
new instance
setProxy(MonitorProxy)
requestStarts
initialize(MonitorProxy)
return Monitor
setTimerTrigger()
setTimerTrigger()
do whatever necessary

Visual Programming

cosylab
CONTROL SYSTEM LABORATORY
www.cosylab.com

- Panels in RAD (no hand-written code)
  - implement default behavior
    - notifications to user when communication errors occur
      - timeouts
      - exceptions
      - interpret error numbers
    - callback, monitor and alarm queued&dispatched as event
    - default lifecycle management
      - initialization when GUI becomes visible
      - connection when all necessary parameters have been set
      - destruction when application closes
  - ServiceBean provides access to services in “visual programming” mode

Serious Programming

- Manual programming with Abean device Beans
  - device manual = API (Bean properties & methods)
  - compile-time error checking through strong typing
  - sync and async program flow (through lock object)
  - group Abeans into families for same behaviour
  - fine-grained control:~10 event types, ~15 system properties
- Abeans discover vis/man mode => change behaviour
- Aggregate data for display of large number of similar devices (i.e. tables)