CAFE
A MODERN C++ INTERFACE TO THE EPICS CHANNEL ACCESS LIBRARY

J. Chrin, M.C. Sloan, Paul Scherrer Institut, 5232 Villigen PSI, Switzerland

ABSTRACT

CAFE (Channel Access interFacE) is a C++ library that provides a modern, multifaceted interface to the EPICS-based control system. CAFE makes extensive use of templates and containers with multiple STL-compatible access methods to enhance efficiency, flexibility and performance. Stability and robustness are accomplished by ensuring that connectivity to EPICS channels remains in a well defined state in every eventuality, and results of all synchronous and asynchronous operations are captured and reported with integrity. CAFE presents the user with a number of options for writing and retrieving data to and from the control system. In addition to basic read and write operations, a further abstraction layer provides transparency to more intricate functionality involving logical sets of data; such object sequences are easily instantiated through an XML-based configuration mechanism. CAFE's suitability for use in a broad spectrum of applications is demonstrated. These range from high performance Qt GUI (Graphical User Interface) control widgets, to event processing agents that propagate data through the Object Managements Group's Data Distribution Service (OMG-DDS), to script-like frameworks such as MATLAB. The methodology for the modular use of CAFE serves to improve maintainability by enforcing a logical boundary between the channel access components and the programming extensions of the application framework at hand.

CAFE À LA MAISON

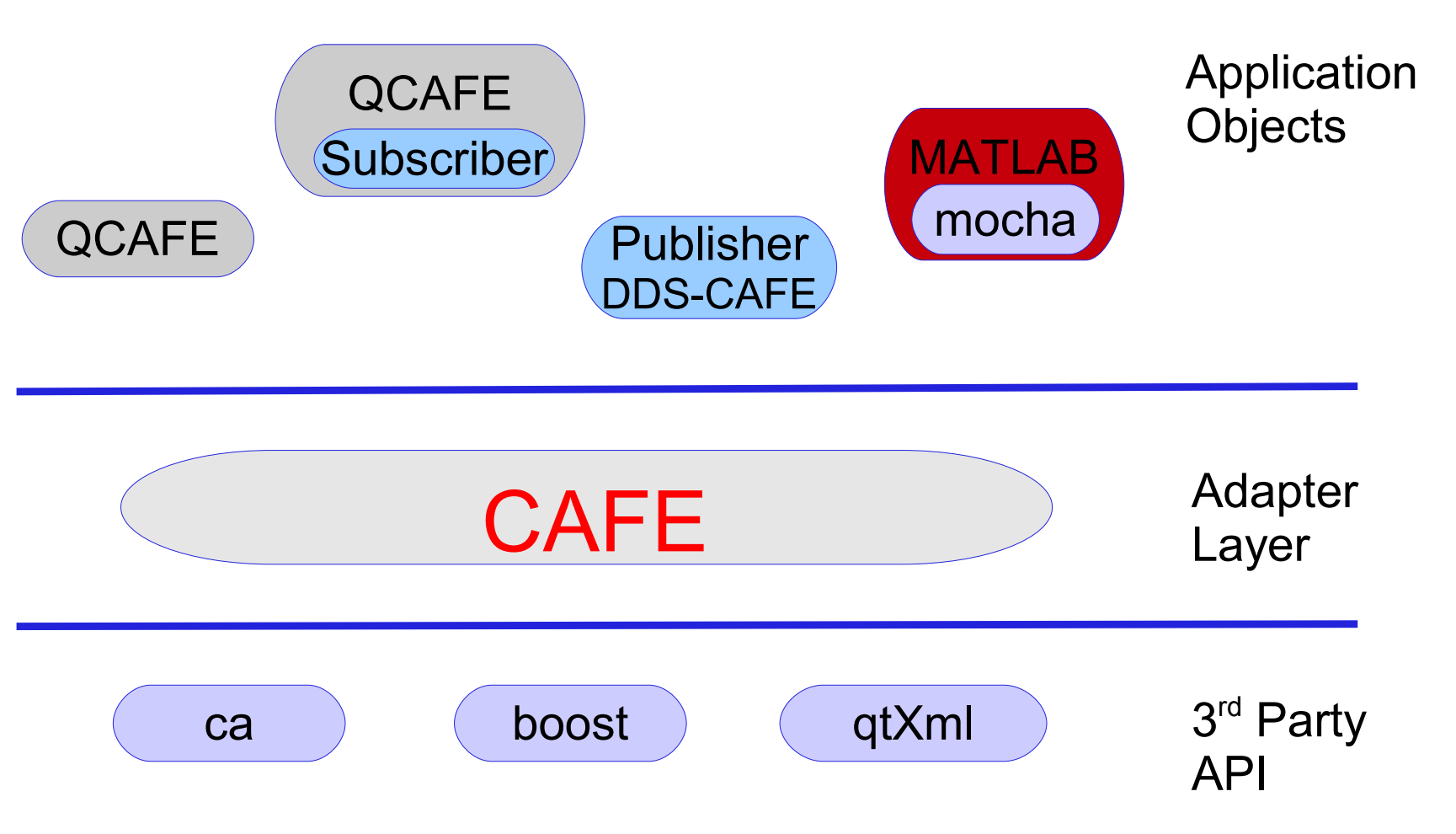
CAFE Implementation

- Synchronous, asynchronous, blocking, non-blocking, interaction for individual, collections/groups (configured through XML)
- Data passed through native type but may be presented to client in any meaningful type (true also of enumerated types)
- Callback functions for CA connection, event, access right handlers
- Boost multi-index containers for memory resident PV data objects
- STL-like access methods for fast retrieval/modification of container elements
- Extensive use of templates, CA details well hidden from user

CAFE Library Advantages

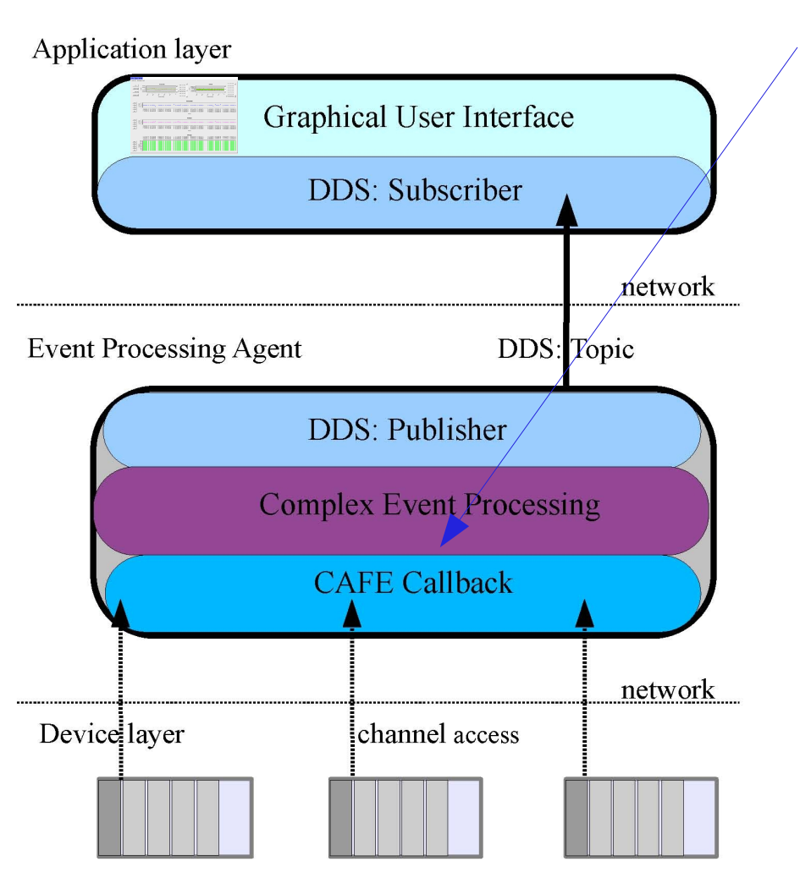
- Inherent simplicity and convenience of maintaining CA interface code in a single location
- New CA functionalities in future EPICS releases readily integrated
- Simplifies interface to other domains
- In-house CA client expertise ensures a quick response to problem solving

Application objects connecting to CAFE's adapter to channel access



DDS-CAFE

Data Distribution Service (DDS) is a recent OMG standard: supports high performance publish-subscribe messaging with configurable Quality-of-Service guarantees



Event Processing Agent (EPA) aggregates, verifies, analyzes low-level DBPM data and distributes summarized results through DDS to the GUI. The EPA uses CAFE to establish a callback mechanism to EPICS:

groupName='gDBPM' defined in EPA XML-configuration file

```
long CAFE::openGroup(char * groupName,
        unsigned long &grouphandle)  throw (CAFEException);

long CAFE::monitorGroup (unsigned long  groupHandle,
PVGroup &pvgroup,  pCallback userCallbackHandlerMonitor);
```

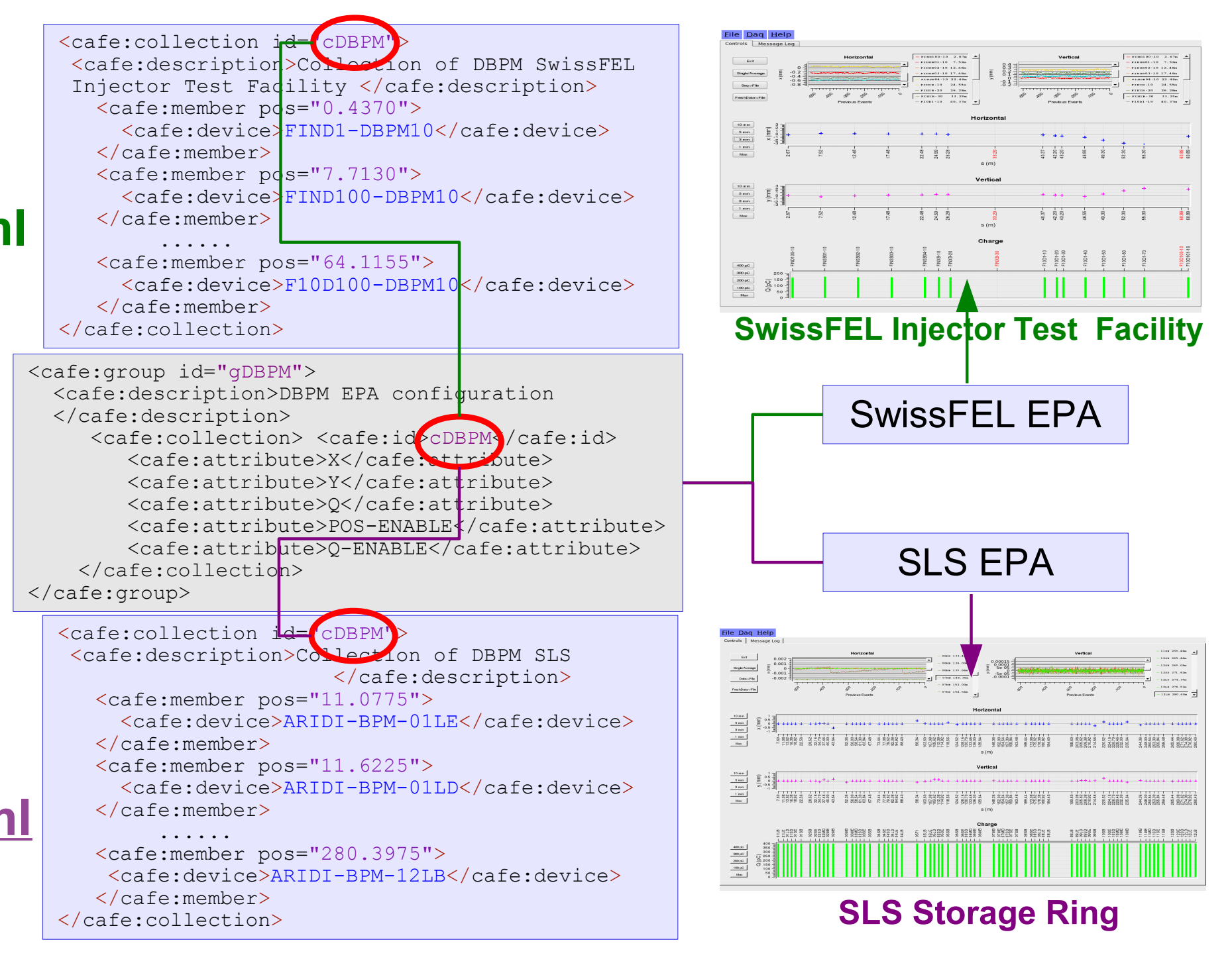
Universal Representation of Particle Accelerators in XML
J. Chrin, R. Krempaska, H. Lutz, G. Prekas (PSI), T. Pelaia (ORNL)
IPAC 2011, p. 2275

EPAs are configured dynamically through XML.
Applications may be adapted to changes in the accelerator system by simple modification to the configuration files.

./xmlFilter SwissFEL.xml
node=DBPM >
CAFECollection_SwissFEL.xml

EPA_DBPM.xml

./xmlFilter SLS.xml
node=DBPM >
CAFECollection_SLS.xml



```
<cafe:collection id='cDBPM'>
 <cafe:description>Collection of DBPM SwissFEL
 Injector Test Facility </cafe:description>
   <cafe:member pos="0.4370">
     <cafe:device>FIND1-DBPM10</cafe:device>
   </cafe:member>
   <cafe:member pos="7.7130">
     <cafe:device>FIND100-DBPM10</cafe:device>
   </cafe:member>
       ......
   <cafe:member pos="64.1155">
     <cafe:device>F10D100-DBPM10</cafe:device>
   </cafe:member>
</cafe:collection>
```

```
<cafe:group id="gDBPM">
  <cafe:description>DBPM EPA configuration
  </cafe:description>
     <cafe:collection> <cafe:id>cDBPM</cafe:id>
        <cafe:attribute>X</cafe:attribute>
        <cafe:attribute>Y</cafe:attribute>
        <cafe:attribute>Q</cafe:attribute>
        <cafe:attribute>POS-ENABLE</cafe:attribute>
        <cafe:attribute>Q-ENABLE</cafe:attribute>
    </cafe:collection>
</cafe:group>
```

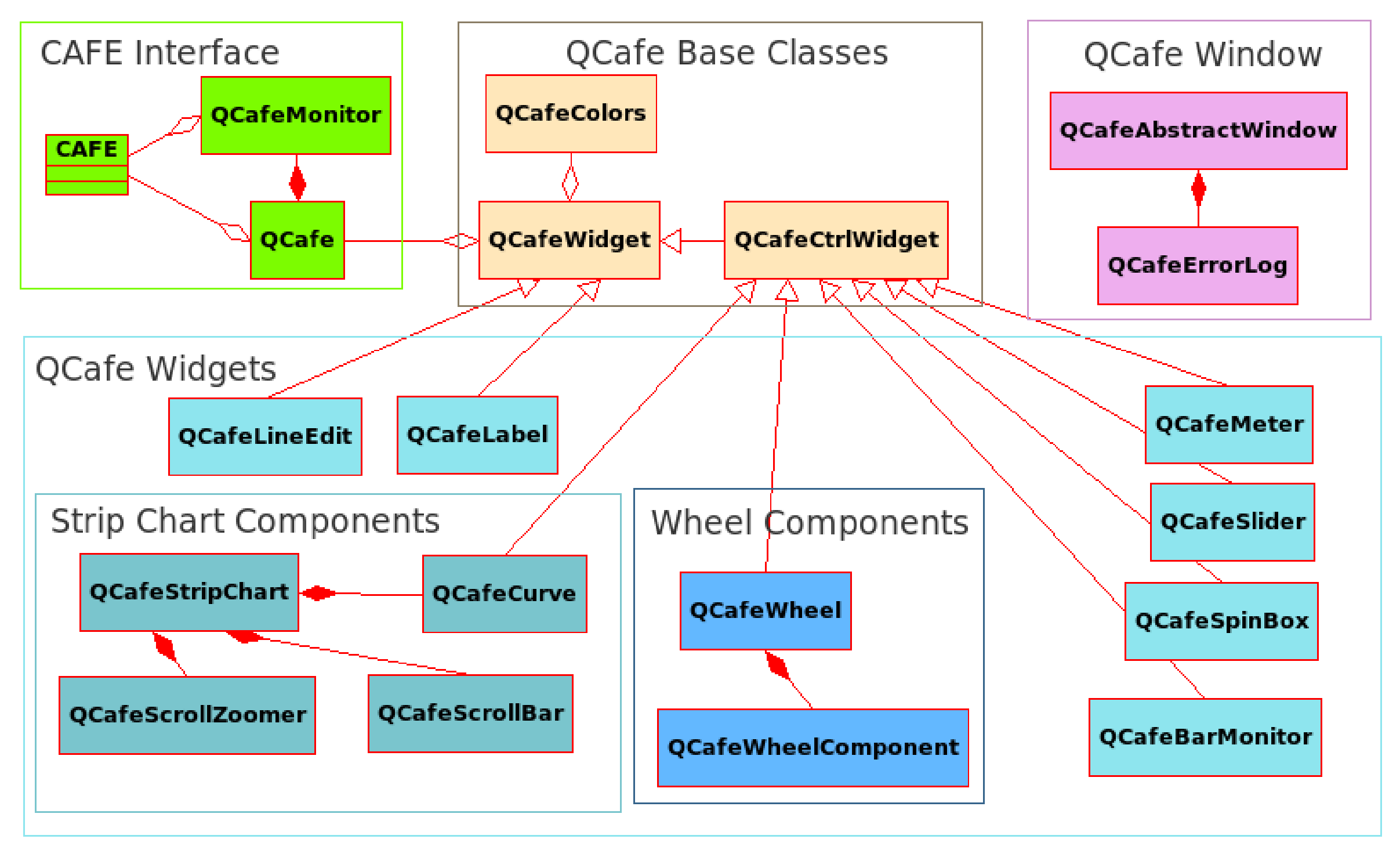
```
<cafe:collection id='cDBPM'>
 <cafe:description>Collection of DBPM SLS
                                </cafe:description>
   <cafe:member pos="11.0775">
     <cafe:device>ARIDI-BPM-01LE</cafe:device>
   </cafe:member>
   <cafe:member pos="11.6225">
     <cafe:device>ARIDI-BPM-01LD</cafe:device>
   </cafe:member>
       ......
   <cafe:member pos="280.3975">
    <cafe:device>ARIDI-BPM-12LB</cafe:device>
   </cafe:member>
</cafe:collection>
```

QCafe

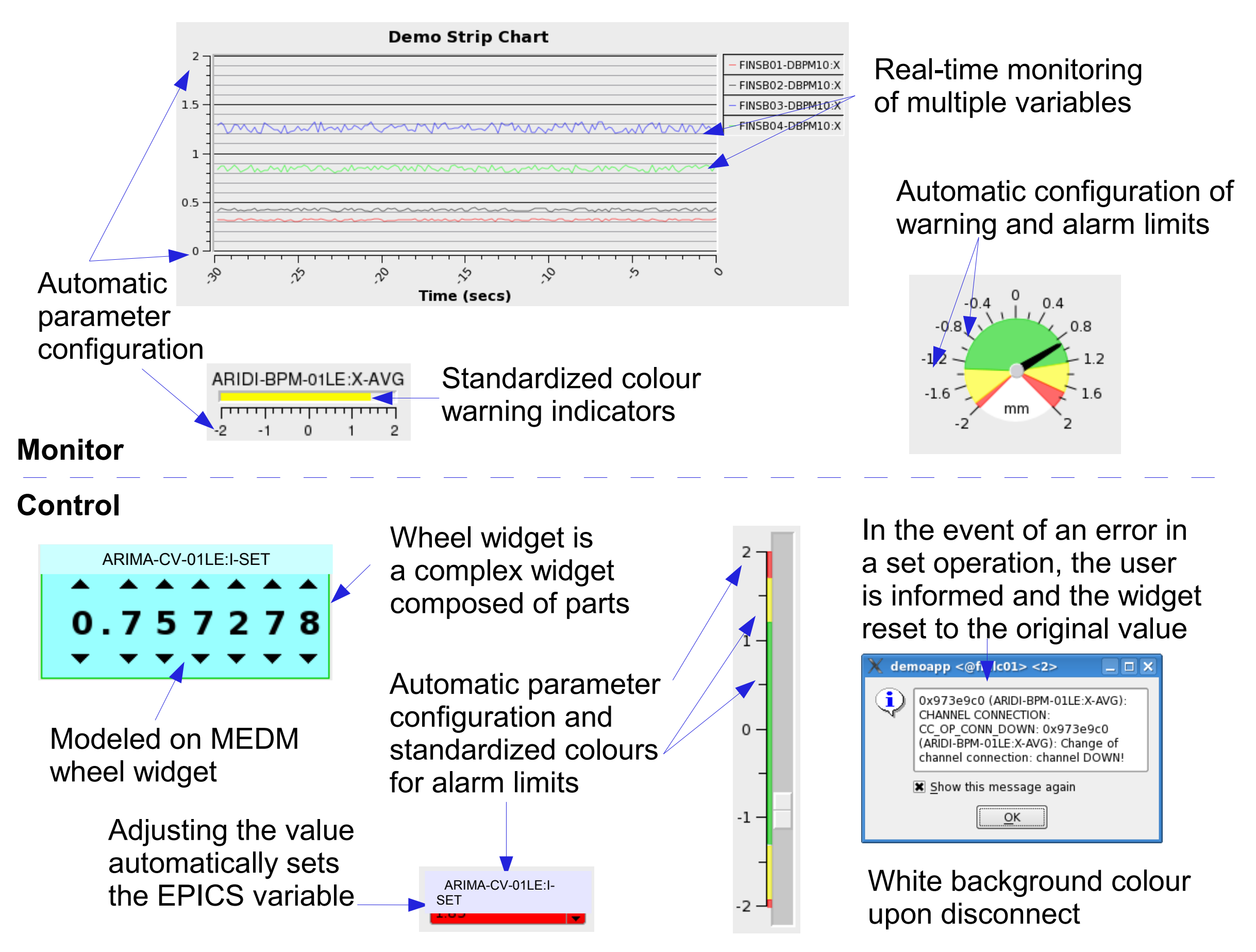
QCafe = Qt + CAFE

Speed, flexibility and power of C++/Qt combined with independent and robust Channel Access using CAFE

QCafe Class Diagram



QCafe Monitor and Control Widgets


Demo Strip Chart
FINSB01-DBPM10:X
FINSB02-DBPM10:X
FINSB03-DBPM10:X
FINSB04-DBPM10:X
Time (secs)
Real-time monitoring of multiple variables
Automatic configuration of warning and alarm limits
Automatic parameter configuration
ARIDI-BPM-01LE:X-AVG
Standardized colour warning indicators
mm
Monitor
Control
ARIMA-CV-01LE:I-SET
Wheel widget is a complex widget composed of parts
Automatic parameter configuration and standardized colours for alarm limits
In the event of an error in a set operation, the user is informed and the widget reset to the original value
demoapp
Show this message again
OK
Modeled on MEDM wheel widget
Adjusting the value automatically sets the EPICS variable
White background colour upon disconnect

```
0x973e9c0 (ARIDI-BPM-01LE:X-AVG):
CHANNEL CONNECTION:
CC_OP_CONN_DOWN: 0x973e9c0
(ARIDI-BPM-01LE:X-AVG): Change of
channel connection: channel DOWN!
```

mocha

mocha: MATLAB Objects for Channel Access

- Access to all MATLAB primitive data types (analysis of camera data)
- MATLAB oo mocha-aware classes, though intensive computation moved to mocha MEX-file for enhanced performance (orbit / machine data)
  Classes, on initialization, extract static data, i.e. node names, positions from the accelerator XML
- Compilation on a 64-bit Linux server (machine snapshot / 3D simulations)

mocha monitors:

- Cache updated values, optionally execute a script to update widget (but MATLAB's Event Listener model may be a better approach)

HARD ROCK CAFE

Acceptance tests from several external APIs have been performed and give us confidence that the CAFE library is in a credible, 'Hard as Rock', state and is ready for deployment. Further developments with QCafe will lead to superior high performance GUI applications, while integration to Qt Designer will allow control applications to be developed without having to write code. The DDS-CAFE EPAs allow for analyzed low-level data to be delivered to clients through a state-of-the-art publish-subscribe mechanism, thereby catering for a reactive form of programming. CAFE's mocha API has demonstrated tangible benefits to application developers, and work towards its consolidation draws near. The approach of having an extensive and flexible, C++ channel access library to serve as a host API for C/C++ based language extensions leads to a thoroughly tested code base for several declarative and domain-specific languages, ultimately reducing and simplifying maintenance of code. If proved effective, then "Hard Rock Cafe's" mission, "to love all and serve all", is one that may be equally adopted by our very own brand of "Hard Rock CAFE".