XDAQ

Contents

1 Introduction

Please find here the official documentation: https://svnweb.cern.ch/trac/cmsos

XDAQ is a framework designed specifically for the development of distributed data acquisition systems. The distributed programming environment follows a layered middleware approach. Middleware services include information dispatching to applications, data transmission, exception handling facilities, access to configuration parameters, and location lookup (address resolution) of applications and services. XDAQ relies on two high-level protocols: I2O for binary messaging and SOAP for interfacing to external systems.

2 Installation

- SLC4 or SLC5 (32bits)

```
Linux localhost.localdomain 2.6.18-274.7.1.el5 #1 SMP Mon Oct 24 11:17:22 CEST 2011 i686 i686 i
```

- YUM client configuration: */etc/yum.repos.d/xdaq.repo*

```
[xdaq]
name=XDAQ Software Base
baseurl=https://xdaq.web.cern.ch/xdaq/repo/10/slc4x/i386/base/RPMS/
enabled=1
gpgcheck=0

[xdaq-sources]
name=XDAQ Software Base Sources
baseurl=https://xdaq.web.cern.ch/xdaq/repo/10/slc4x/i386/base/SRPMS/
enabled=1
gpgcheck=0

[xdaq-updates]
name=XDAQ Software Updates
baseurl=https://xdaq.web.cern.ch/xdaq/repo/10/slc4x/i386/updates/RPMS/
enabled=1
gpgcheck=0

[xdaq-updates-sources]
name=XDAQ Software Updates Sources
baseurl=https://xdaq.web.cern.ch/xdaq/repo/10/slc4x/i386/updates/SRPMS/
enabled=1
gpgcheck=0

[xdaq-extras]
name=XDAQ Software Extras
baseurl=http://xdaq.web.cern.ch/xdaq/repo/10/slc4x/i386/extras/RPMS/
enabled=1
gpgcheck=0

[xdaq-test]
name=XDAQ Software Test
baseurl=http://xdaq.web.cern.ch/xdaq/repo/10/slc4x/i386/test/RPMS/
enabled=0
gpgcheck=0

[xdaq-test-sources]
name=XDAQ Software Test Sources
baseurl=http://xdaq.web.cern.ch/xdaq/repo/10/slc4x/i386/test/SRPMS/
enabled=0
gpgcheck=0
```

- Installation

```
# yum groupinstall extern_coretools
# yum groupinstall coretools
# yum groupinstall extern_powerpack
# yum groupinstall powerpack

# rpm -qa | grep daq
```

- Configuration */.bashrc*

```
export XDAQ_ROOT=/opt/xdaq
```

```
export XDAQ_DOCUMENT_ROOT=/opt/xdaq/htdocs
export LD_LIBRARY_PATH=/opt/xdaq/lib
```

- Running XDAQ

```
# /etc/init.d/httpd start
$ $XDAQ_ROOT/bin/xdaq.exe
```

Now we should access XDAQ process at http://localhost:1972

3 Hello World exemples

3.1 Compilation

- */xdaq/test/hello/include/hello/HelloWorld.h*

```
#ifndef _hello_HelloWorld_h_
#define _hello_HelloWorld_h_

#include "xdaq/Application.h"

namespace hello {
class HelloWorld: public xdaq::Application
{
  public:

    XDAQ_INSTANTIATOR();

    HelloWorld(xdaq::ApplicationStub * s)
      throw (xdaq::exception::Exception);
};
}

#endif
```

- */xdaq/test/hello/src/common/HelloWorld.cc*

```
#include "hello/HelloWorld.h"

XDAQ_INSTANTIATOR_IMPL(hello::HelloWorld)

hello::HelloWorld::HelloWorld(xdaq::ApplicationStub * s)
  throw (xdaq::exception::Exception): xdaq::Application(s)
{
  LOG4CPLUS_INFO(this->getApplicationLogger(),"Hello World!");
}
```

- */xdaq/test/hello/include/hello/version.h*

```
#ifndef _hello_version_h_
#define _hello_version_h_

#include "config/PackageInfo.h"
```

```
#define MYPACKAGE_VERSION_MAJOR 1
#define MYPACKAGE_VERSION_MINOR 0
#define MYPACKAGE_VERSION_PATCH 0
#undef MYPACKAGE_PREVIOUS_VERSIONS

#define MYPACKAGE_VERSION_CODE PACKAGE_VERSION_CODE(MYPACKAGE_VERSION_MAJOR, MYPACKAGE_VERSION_
        MYPACKAGE_VERSION_PATCH)
#ifndef MYPACKAGE_PREVIOUS_VERSIONS
#define MYPACKAGE_FULL_VERSION_LIST  PACKAGE_VERSION_STRING(MYPACKAGE_VERSION_MAJOR, \
        MYPACKAGE_VERSION_MINOR, MYPACKAGE_VERSION_PATCH)
#else
#define MYPACKAGE_FULL_VERSION_LIST MYPACKAGE_PREVIOUS_VERSIONS "," \
        PACKAGE_VERSION_STRING(MYPACKAGE_VERSION_MAJOR, MYPACKAGE_VERSION_MINOR, MYPACKAGE_VERS
#endif

namespace hello
{
const std::string package  =  "hello";
const std::string versions = MYPACKAGE_FULL_VERSION_LIST;
const std::string summary = "Hello World Example";
const std::string description = "A simple XDAQ application";
const std::string authors = "Roland and Luciano";
const std::string link = "http://xdaq.web.cern.ch";
config::PackageInfo getPackageInfo();
void checkPackageDependencies() throw (config::PackageInfo::VersionException);
std::set<std::string, std::less<std::string> > getPackageDependencies();
}

#endif
```

- */xdaq/test/hello/src/common/version.cc*

```
#include "config/version.h"
#include "xcept/version.h"
#include "xdaq/version.h"
#include "hello/version.h"

GETPACKAGEINFO(hello)

void hello::checkPackageDependencies() throw (config::PackageInfo::VersionException)
{
CHECKDEPENDENCY(config);
CHECKDEPENDENCY(xcept);
CHECKDEPENDENCY(xdaq);
}

std::set<std::string, std::less<std::string> > hello::getPackageDependencies()
{
std::set<std::string, std::less<std::string> > dependencies;

ADDDEPENDENCY(dependencies,config);
ADDDEPENDENCY(dependencies,xcept);
ADDDEPENDENCY(dependencies,xdaq);
```

```
return dependencies;
}
```

- */xdaq/test/hello/Makefile*

```
BUILD_HOME:=$(shell pwd)/../..

include $(XDAQ_ROOT)/config/mfAutoconf.rules
include $(XDAQ_ROOT)/config/mfDefs.$(XDAQ_OS)

Project=test
Package=hello

Sources= HelloWorld.cc version.cc

DynamicLibrary=hello

include $(XDAQ_ROOT)/config/Makefile.rules
include $(XDAQ_ROOT)/config/mfRPM.rules
```

- Compiling

```
# yum install gcc-c++ e2fsprogs-devel
$ export XDAQ_ROOT=/opt/xdaq
$ cd ~/xdaq/test/hello
$ make
$ ls ~/xdaq/test/hello/lib/linux/x86_sl5/libhello.so
```

3.2 Running

- */xdaq/test/hello/xml/hello.xml*

```
<?xml version='1.0'?>
<xc:Partition xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xc="http://xdaq.web.cern.ch/xdaq/xsd/2004/XMLConfiguration-30">

  <!-- Declare a context that contain applcation -->
  <xc:Context url="http://localhost:1972">

    <!-- Declare a HelloWorld application -->
    <xc:Application class="hello::HelloWorld" id="254" instance="0" network="local"/>

    <!-- Shared object library that contains the HelloWorld inplementation -->
    <xc:Module>${HOME}/xdaq/test/hello/lib/linux/x86_slc5/libhello.so</xc:Module>

  </xc:Context>

</xc:Partition>
```

- Running

```
$ export LD_LIBRARY_PATH=/opt/xdaq/lib
$ /opt/xdaq/bin/xdaq.exe -p 1972 -c ~/xdaq/test/hello/xml/hello.xml -l DEBUG
...
22 Nov 2011 14:34:00.914 [3086358272] INFO  localdomain.localhost.p:1972.hello::HelloWorld.inst
...
```

3.3 Web browser access

- */xdaq/test/hello/include/hello/SimpleWeb.h*

```
#ifndef _hello_SimpleWeb_h
#define _hello_SimpleWeb_h

#include "xdaq/Application.h"
#include "xgi/Method.h"

namespace hello {

class SimpleWeb: public xdaq::Application {
  public:

  XDAQ_INSTANTIATOR();

  SimpleWeb(xdaq::ApplicationStub * s) throw (xdaq::exception::Exception);
  void Default(xgi::Input * in, xgi::Output * out ) throw (xgi::exception::Exception);
};
}
#endif
```

- */xdaq/test/hello/src/common/SimpleWeb.cc*

```
#include "cgicc/HTMLDoctype.h"
#include "cgicc/HTMLClasses.h"
#include "hello/SimpleWeb.h"

XDAQ_INSTANTIATOR_IMPL(hello::SimpleWeb)

hello::SimpleWeb::SimpleWeb(xdaq::ApplicationStub * s)
  throw (xdaq::exception::Exception): xdaq::Application(s)
{
    xgi::bind(this,&SimpleWeb::Default, "Default");
}

void hello::SimpleWeb::Default(xgi::Input * in, xgi::Output * out )
  throw (xgi::exception::Exception)
{
  *out << cgicc::HTMLDoctype(cgicc::HTMLDoctype::eStrict) << std::endl;
  *out << cgicc::html().set("lang", "en").set("dir","ltr") << std::endl;
  *out << cgicc::title("Simple Web") << std::endl;
  *out << cgicc::a("Visit the XDAQ Web site").set("href","http://xdaq.web.cern.ch") << endl;
}
```

- */xdaq/test/hello/Makefile*

```
BUILD_HOME:=$(shell pwd)/../..

include $(XDAQ_ROOT)/config/mfAutoconf.rules
include $(XDAQ_ROOT)/config/mfDefs.$(XDAQ_OS)

Project=test
Package=hello
```

```
Sources= HelloWorld.cc version.cc SimpleWeb.cc

DynamicLibrary=hello

include $(XDAQ_ROOT)/config/Makefile.rules
include $(XDAQ_ROOT)/config/mfRPM.rules
```

- */xdaq/test/hello/xml/hello.xml*

```
<?xml version='1.0'?>
<xc:Partition xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xc="http://xdaq.web.cern.ch/xdaq/xsd/2004/XMLConfiguration-30">

  <!-- Declare a context that contain applcation -->
  <xc:Context url="http://localhost:1972">

    <!-- Declare a HelloWorld application -->
    <xc:Application class="hello::HelloWorld" id="254" instance="0" network="local"/>
    <xc:Application class="hello::SimpleWeb" id="255" instance="0" network="local"/>

    <!-- Shared object library that contains the HelloWorld inplementation -->
    <xc:Module>${HOME}/xdaq/test/hello/lib/linux/x86_slc5/libhello.so</xc:Module>

  </xc:Context>

</xc:Partition>
```

- Compiling and running

```
$ export XDAQ_ROOT=/opt/xdaq
$ cd ~/xdaq/test/hello
$ make

$ export LD_LIBRARY_PATH=/opt/xdaq/lib
$ /opt/xdaq/bin/xdaq.exe -p 1972 -c ~/xdaq/test/hello/xml/hello.xml -l DEBUG
```

3.4 Properties

- */xdaq/test/hello/include/hello/MyApplication.h*

```
#ifndef _hello_MyApplication_h
#define _hello_MyApplication_h

#include "xdaq/Application.h"

#include "xdata/ActionListener.h"
#include "xdata/UnsignedLong.h"
#include "xdata/UnsignedInteger.h"
#include "xdata/Vector.h"

namespace hello {
```

```
  class MyApplication: public xdaq::Application, xdata::ActionListener {
  public:

  XDAQ_INSTANTIATOR();

  MyApplication(xdaq::ApplicationStub * s) throw (xdaq::exception::Exception);

  void actionPerformed (xdata::Event& e);

  protected:

  xdata::UnsignedLong myCounter_;
  xdata::Vector<xdata::UnsignedLong> myVector_;

};
}
#endif
```

- */xdaq/test/hello/src/common/MyApplication.cc*

```
#include "hello/MyApplication.h"

XDAQ_INSTANTIATOR_IMPL(hello::MyApplication)

hello::MyApplication::MyApplication (xdaq::ApplicationStub * s)
  throw (xdaq::exception::Exception): xdaq::Application(s)
{
  // initialize to a default value
  myCounter_ = 0;

  // make Variables available in the application InfoSpace
  this->getApplicationInfoSpace()->fireItemAvailable("myCounter", &myCounter_);
  this->getApplicationInfoSpace()->fireItemAvailable("myVector", &myVector_);

  // Detect when the setting of default parameters has been performed
  this->getApplicationInfoSpace()->addListener(this, "urn:xdaq-event:setDefaultValues");
}

void hello::MyApplication::actionPerformed (xdata::Event& e)
{
  if ( e.type() == "urn:xdaq-event:setDefaultValues" )
  {
    std::stringstream ss;

    ss << "myCounter=[" << myCounter_ << "]" << std::endl;

    for ( size_t i = 0 ; i != myVector_.size() ; i++ )
    {
      ss << "myVector=[" << i << "]=[" << myVector_[i] << "]" << std::endl;
    }
    LOG4CPLUS_INFO(this->getApplicationLogger(), ss.str());
  }
}
```

- */xdaq/test/hello/xml/hello.xml*

```
<?xml version='1.0'?>
<xc:Partition xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xc="http://xdaq.web.cern.ch/xdaq/xsd/2004/XMLConfiguration-30">

  <!-- Declare a context that contain applcation -->
  <xc:Context url="http://localhost:1972">

    <!-- Declare a HelloWorld application -->
    <xc:Application class="hello::HelloWorld" id="254" instance="0" network="local"/>
    <xc:Application class="hello::SimpleWeb" id="255" instance="0" network="local"/>

    <xc:Application class="hello::MyApplication" id="2" instance="0" network="local">
      <properties xmlns="urn:xdaq-application:MyApplication"
          xsi:type="soapenc:Struct">
        <myCounter xsi:type="xsd:unsignedLong">10</myCounter>
        <myVector xsi:type="soapenc:Array" soapenc:arrayType="xsd:ur-type[2]">
  <item xsi:type="xsd:unsignedLong" soapenc:position="[0]">1234</item>
          <item xsi:type="xsd:unsignedLong" soapenc:position="[1]">4321</item>
        </myVector>
      </properties>
    </xc:Application>

    <!-- Shared object library that contains the HelloWorld inplementation -->
    <xc:Module>${HOME}/xdaq/test/hello/lib/linux/x86_slc5/libhello.so</xc:Module>

  </xc:Context>

</xc:Partition>
```

- */xdaq/test/hello/Makefile*

```
BUILD_HOME:=$(shell pwd)/../..

include $(XDAQ_ROOT)/config/mfAutoconf.rules
include $(XDAQ_ROOT)/config/mfDefs.$(XDAQ_OS)

Project=test
Package=hello

Sources= HelloWorld.cc version.cc SimpleWeb.cc MyApplication.cc

DynamicLibrary=hello

include $(XDAQ_ROOT)/config/Makefile.rules
include $(XDAQ_ROOT)/config/mfRPM.rules
```

- Compiling and running

```
$ export XDAQ_ROOT=/opt/xdaq
$ cd ~/xdaq/test/hello
$ make

$ export LD_LIBRARY_PATH=/opt/xdaq/lib
```

```
$ /opt/xdaq/bin/xdaq.exe -p 1972 -c ~/xdaq/test/hello/xml/hello.xml -l DEBUG
...
22 Nov 2011 17:26:51.478 [3086829312] INFO  localdomain.localhost.p:1972.hello::MyApplication.i
myVector=[0]=[1234]
myVector=[1]=[4321]
```

4 Somes tests

4.1 Linking

- */xdaq/test/myClass.hh*

```
#ifndef _LLR_CALICE_LDA_H_
#define _LLR_CALICE_LDA_H_

namespace LLR_CALICE_LDA
{

  class Lda
  {
  public:
    Lda()
    {}

    ~Lda()
    {}
  };
};

#include <ostream>
std::ostream & operator<< (std::ostream & os,
   LLR_CALICE_LDA::Lda const& lda);

#endif /* _LLR_CALICE_LDA_H_ */
```

- */xdaq/test/myClass.cc*

```
#include "myClass.hh"

std::ostream & operator<< (std::ostream & os,
   LLR_CALICE_LDA::Lda const& lda)
{
    os << "Bonjour from lda";
    return os;
}

#if _utMain
#include <iostream>

int main()
{
  using namespace LLR_CALICE_LDA;
```

```
  Lda lda;
  std::cout << lda << std::endl;
}

#endif /* _utMain */
```

- */xdaq/test/makefile*

```
all:            myLib.so \
utMyClass

myClass.o: myClass.cc \
myClass.hh
g++ -c $< -o $@

utMyClass: myClass.cc \
myClass.hh
g++ -D_utMain $< -o $@

myLib.so: myClass.o
g++ -shared $^ -o $@

clean:
rm -f myClass.o
```

- */xdaq/test/hello/src/common/HelloWorld.cc*

```
#include "hello/HelloWorld.h"
#include "../../myClass.hh"

XDAQ_INSTANTIATOR_IMPL(hello::HelloWorld)

hello::HelloWorld::HelloWorld(xdaq::ApplicationStub * s)
  throw (xdaq::exception::Exception): xdaq::Application(s)
{
  LLR_CALICE_LDA::Lda lda;

  printf("coucou\n");
  LOG4CPLUS_INFO(this->getApplicationLogger(),"Hello World!");

  std::cout << lda << std::endl;
  LOG4CPLUS_INFO(this->getApplicationLogger(), lda);
}
```

- */xdaq/test/hello/xml/hello.xml*

```
<?xml version='1.0'?>
<xc:Partition xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xc="http://xdaq.web.cern.ch/xdaq/xsd/2004/XMLConfiguration-30">

  <!-- Declare a context that contain applcation -->
  <xc:Context url="http://localhost:1972">
```

```
    <!-- Declare a HelloWorld application -->
    <xc:Application class="hello::HelloWorld" id="254" instance="0" network="local"/>

    <!-- Shared object library that contains the HelloWorld inplementation -->
    <xc:Module>${HOME}/xdaq/test/myLib.so</xc:Module>
    <xc:Module>${HOME}/xdaq/test/hello/lib/linux/x86_slc5/libhello.so</xc:Module>

  </xc:Context>
</xc:Partition>
```

- Compiling and running

```
$ cd ~/xdaq/test
$ make
$ ./utMyClass

$ export XDAQ_ROOT=/opt/xdaq
$ cd ~/xdaq/test/hello
$ make

$ export LD_LIBRARY_PATH=/opt/xdaq/lib
$ /opt/xdaq/bin/xdaq.exe -p 1972 -c ~/xdaq/test/hello/xml/hello.xml -l DEBUG
...
23 Nov 2011 11:43:40.231 [3086501632] INFO  localdomain.localhost.p:1972.hello::HelloWorld.inst
Bonjour from lda
23 Nov 2011 11:43:40.234 [3086501632] INFO  localdomain.localhost.p:1972.hello::HelloWorld.inst
```

4.2 Using 2 hosts

-

5 Somes others tests

5.1 Threads (todo)

-

5.2 Memory pools (todo)

-

5.3 Message passing (todo)

-

5.4 Package versioning (todo)

-

5.5 Exceptions (todo)

-

5.6 Monitoring (todo)

-

5.7 Data handling (todo)

-

5.8 Web programming (todo)

-

5.9 OS wrapper (todo)

-

5.10 State machine (todo)

-