Introduction to EDNA Framework

Overview of BioSaxs' application of EDNA
Layout

• Presentation of EDNA
  • On-line data analysis framework
  • Create pipeline by chaining plugins
  • Provides many useful building blocks

• What has been implemented for BioSaxs
  • 6 general purpose execution plugins for saxs_* utilities
    • Presentation of datamodels
  • 4 BioSaxs specific plugins
    • Presentation of datamodels
  • 3 parallel applications

• Ongoing developments
  • What is missing ...
  • Priorities for further development.
What is EDNA

• A tool to develop Online Data Analysis programs:
  • Data analysis → as quickly as possible → feedback to users

• A Workflow engine
  • automation of tasks
  • Parallel execution of tasks

• A Reliable tool:
  • Has a strong testing framework
    • Unit tests
    • Execution tests
  • Each plugin IS tested:
    • Test exists
    • Non regression run nightly

• Did I mention the MX community is the key motor for EDNA
Features of EDNA

• EDNA is a robust pipe-lining tool for on-line data analysis
  • It has been tested with thousands of tasks at once

• EDNA allows hi-performances
  • Multi-threaded implementation

• EDNA relies on data-models
  • Visual communication with scientists
  • Automatic bindings with the code

• EDNA has a strong testing framework
  • Unit & Execution tests
  • Non regression test before nightly builds

• EDNA is efficient to program
  • Plugin generator for execution plug-ins based on the data-model
  • Re-use of plug-ins already written (by others)

• EDNA is an international collaboration (with DLS, CCP4, ...)

EDNA provides scientific building blocks

• Execution plugins:
  • MxExecPlugins: 24 exec plugins
  • Exec plugins: 21 exec plugins
    • Saxs(4), SPD (4), FIT2D (2), EDF (2), HDF5 (2), thumbnail, video, …

• Control plugins:
  • MX v1 26 control plugins, 7 exec plugins
  • MX v2 3 control plugins
  • DiffractionCT v1 6 control plugins
  • BioSaxs 5 control plugins

• Other projects (CCP4, Dimple, Darc ...):
  • Managed mainly by Diamond and CCP4
  • 27 plugins: 8 control plugins & 19 execution plugins

• Total: 119 out of 71 Execution plugins, 48 control plugins
Work done for ID14eh3

Reprocess.py → EDNA plugins
Execution plugins

EDNA Execution plugin group
• EDPluginWaitFile-v1.0
  • EDPluginWaitFile
  • EDPluginWaitMultiFile

Reprocess.py uses:
• saxes_mac (normalize)
• saxes_add (apply mask)
• saxes_angle (azimuthal integ.)
• saxes_curves (ascii export)
Example of Datamodels: EDPluginWaitFile-v1.0

- **XSDataInput**:
  - expectedFile: XSDataFile
  - expectedSize: XSDataInteger

- **XSDataResult**:
  - actualFile: XSDataFile
  - actualSize: XSDataInteger
  - timeout: XSDataBoolean

**XSDDataSize** is not applicable here; it is the size of a file in Bytes, not the size of a physical object in meters.

- **XSDataInput**:
  - expectedFile: XSDataFile [{1..*}]
  - expectedSize: XSDataInteger

- **XSDataResult**:
  - actualFile: XSDataFile [{1..*}]
  - actualMinSize: XSDataInteger
  - timeout: XSDataBoolean
Datamodel for bioSaxs

class XDataBioSaxsReprocessv1_0 Model

XDataResult

+ averagedImage: XSDataImage
+ averagedSpectrum: XSDataFile
+ logFile: XSDataFile
+ processLog: XSDataString

XDataInput

+ integratedImage: XSDataImage [1..*]
+ integratedImageSize: XSDataInteger
+ normalizationFactor: XSDataFloat
+ averagedImage: XSDataImage
+ averagedSpectrum: XSDataFile
+ logFile: XSDataFile

XDataBioSaxsSample

+ concentration: XSDataFloat [0..1]
+ comments: XSDataString [0..1]
+ code: XSDataString [0..1]
Implementation for BioSaxs

• **EDPluginBioSaxsNormalizev1_0**: Control plugin for 2 plugins run subsequently:
  • EDPluginWaitFile
  • EDPluginExecSaxsMacv1_0

• **EDPluginBioSaxsAzimutIntv1_0**: Control plugin for 4 plugins run subsequently:
  • EDPluginWaitFile
  • EDPluginExecSaxsAddv1_0
  • EDPluginExecSaxsAnglev1_0
  • EDPluginExecSaxsCurvesv1_0

• **EDPluginBioSaxsAveragev1_0**: Control plugin for 3 plugins run subsequently:
  • EDPluginWaitMultiFile
  • EDPluginExecSaxsMacv1_0
  • EDPluginExecSaxsCurvesv1_0
EDNA Launchers

• As parallel applications (available from PxSoft)
  - NormalizeRaw.py (online & offline)
  - AzimuthalIntegration.py (online & offline)
  - AverageRun.py (only offline)

• As tango server
  - Not yet tested with BioSaxs plugins but should work.
  - Only tango → edna communication

• As a stand-alone EDNA application:
  $ edna-plugin-launcher --execute EDPlugin --inputFile parameters.xml
Work still to be done

• How important is the order of the headers in your files
  • If important: some work is needed
  • Carriage return is OS dependent (removed systematic \r\n)

• Reprocess is not (yet?) re-implemented
  • But is it needed ?
  • Input interface not re-written

• Communication back to spec is not really satisfactory
  • Replace the use of SpecVariable by edna→tango communication
A Light for Science

European Synchrotron Radiation Facility