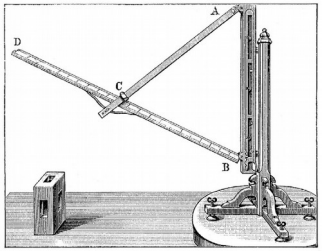
Triquetrum

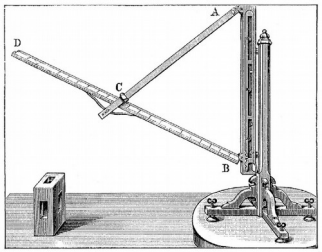
integrating workflows in scientific software

Erwin De Ley, iSencia & Christopher Brooks, UC Berkeley



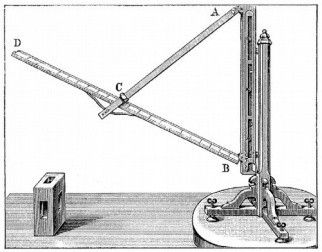
Agenda

- Intro - Scientific software context
- Triquetrum overview
- Workflow features
- Task processing
- Integration approaches
- Project results
- Roadmap



What is Triquetrum?

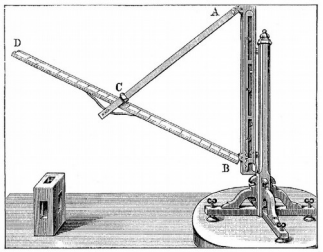
- Triquetrum is an **Eclipse project** that uses the **Ptolemy II** actor-oriented execution engine to provide **run time semantics** for use in **workflows**.
- The project started in 2015 as a project in the Eclipse **Science Working Group**.
- Triquetrum is named for the three sided astronomical instrument that Mr. Ptolemy is holding.
- Pronounced tri-QUET-rum not ~~tri-QUEET-rum~~



Workflows?

Sequence of activities to achieve a certain result

- Pre-defined or ad-hoc?
- Explicitly defined models or implicit in application logic or UI?
- Repeatable?
- Interactive or (semi-)automated?



Benefits of workflow systems

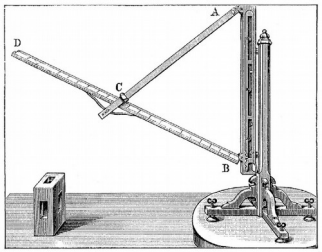
Graphical executable models

- Eases collaboration between stakeholders with different skills
- Self-documenting

Encapsulate technical features

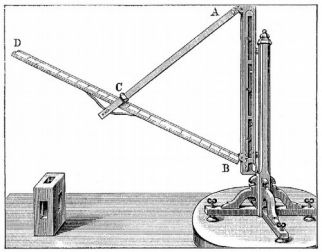
- Automated provenance / tracing
- Consistent error handling
- Concurrent processing, high performance computing
- Integration libraries & much more : security, versioning, scheduling,...

Promotes separation of concerns for software development, model design, process execution, support



Sample applications

- Process control for scientific experiments
  - Data acquisition
  - Equipment control
  - Integrated error recognition and recovery
  - Monitoring & alarming
- (Semi-)automated data reduction and analysis
- Soft real-time feedback between control & analysis in integrated workflows!
- Interactive assistance / support automation
- ...



Context for scientific software

Software systems are crucial in many scientific disciplines

Experiments should be repeatable and reproducable

Software tools & models are part of an experiment's “protocol”

Increasing complexity of experimental devices

Increasing detector speeds and data volumes

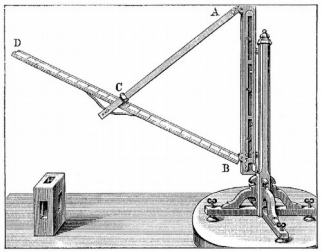
Demands for increasing utilization and efficiency of high-cost equipment

Less time for repetitive & ever-more-complex set-up cycles

Ever-growing rates and volumes of scientific data-sets, combined with requirements for fast and robust processing

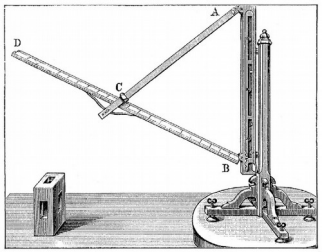
No longer only a situation at BIG science/institutes

The integration of a workflow system in a modular scientific software platform, combining data- and process-management, can bring many benefits



Agenda

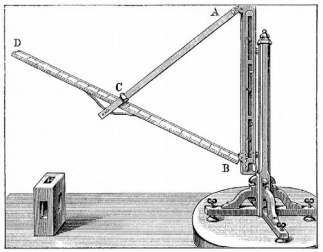
- Intro - Scientific software context
- Triquetrum overview
- Workflow features
- Task processing
- Integration approaches
- Project results
- Roadmap



Triquetrum Goals

- Deliver an **open platform** for managing & executing **workflows**
- Designed for **integration**
- Provide **extension** APIs & services, focus on scientific software

- Support a wide range of use cases:
  - Automated processes based on predefined models
  - Replaying ad-hoc research workflows
    based on a recording of user interactions
  - Allow users to define and execute small and large models



Triquetrum is building on...

Integration of Ptolemy II in an Eclipse and OSGi technology stack.

Ptolemy II (Berkeley, BSD License):

“Ptolemy II is an open-source software framework supporting experimentation with actor-oriented design.”

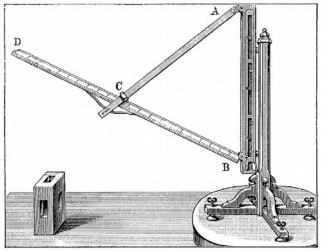
Workflow- and Task-oriented features from Passerelle.

Passerelle is an eclipselabs project, using Ptolemy II as its process engine.
It has been applied since 2004 as a workflow solution:

- At synchrotrons for automated control & data acquisition and data analysis.
- As automated diagnostic engine for repair and customer support in telecoms.

Set of frameworks and technologies of the Eclipse Foundation.

Equinox, Graphiti, EMF, RCP, ...



System overview

Domain-specific workflow actors

Extra actors domain A

Extra actors domain B

Extra actors domain ...

Remote services

Workflow repository service REST facade

Workflow execution service REST facade

Task processing broker REST facade

Triquetrum RCP

Workflow editor

Admin views

Custom forms

Diagram EMF model

Task processing API

Workflow Execution Service

Workflow Repository service

Services layer

Ptolemy II actors

Ptolemy II SDF MoC *

Ptolemy II PN MoC *

Ptolemy II core

Persistent execution tracing

Remoting connectors

Graphiti

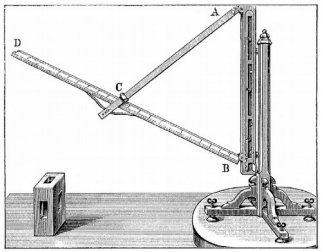
EMF Forms

GEF3

EMF

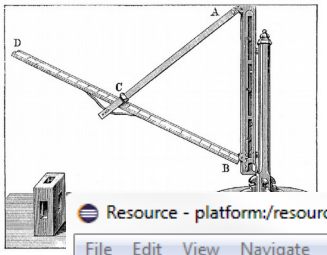
Support modules

* MoC : Model of Computation



Agenda

- Intro - Scientific software context
- Triquetrum overview
- Workflow features
- Task processing
- Integration approaches
- Project results
- Roadmap



Workflow editor

Resource - platform:/resource/test/workflows/ICEdemo1.tdml#/0 - Eclipse Platform

File Edit View Navigate Search Project Run Window Help

Pro... Palette ICEdemo1

test
workflows
ICEdemo1
lissajous.t
vertexTest
ICEdemo1.tdr

type filter text

Utilities
Annotation
Parameter
StringParameter
Vertex
Directors
Sources
Constant
Ramp
IO
CSVReader
Sinks
Submodels
Test
Flow control
Transformation
AddSubtract
MovingAverage
TrigFunction
Examples

SDF director
filePath : C:\data\triquetru...
smoothingFactor : 10
enableBackwardTypeInfer...
_defaultInferredWidthTo1 :...
CSV
$\frac{1}{N}\sum_{i=0}^{N-1} X_{n-i}$
Smoothed ...

Outline

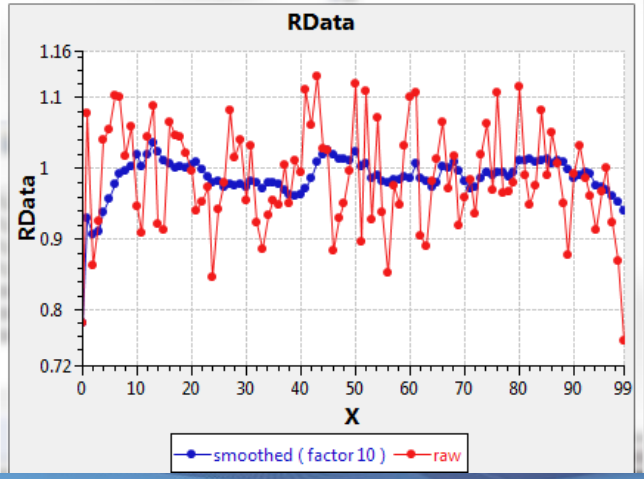
Tasks Console History Problems

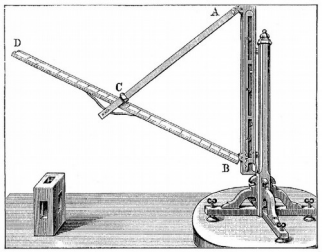
Triq Console

```
2016-05-24 21:40:55,406 TRACE [main]  svg.SvgModelElementShape (f
2016-05-24 21:40:55,408 TRACE [main]  svg.SvgModelElementShape (f
2016-05-24 21:40:55,409 TRACE [main]  svg.SvgModelElementShape (f
2016-05-24 21:40:55,410 TRACE [main]  svg.SvgModelElementShape (f
2016-05-24 21:40:55,412 TRACE [main]  ptolemy.PtolemyModelElement
2016-05-24 21:40:55,412 DEBUG [main]  ptolemy.PtolemyModelElement
2016-05-24 21:40:55,413 TRACE [main]  ptolemy.PtolemyModelElement
```

83M of 479M

Smoothed data plot



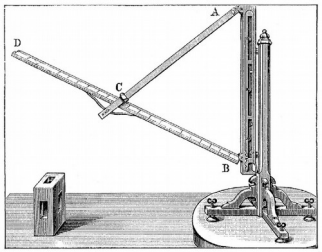


Workflow editor

developed using...

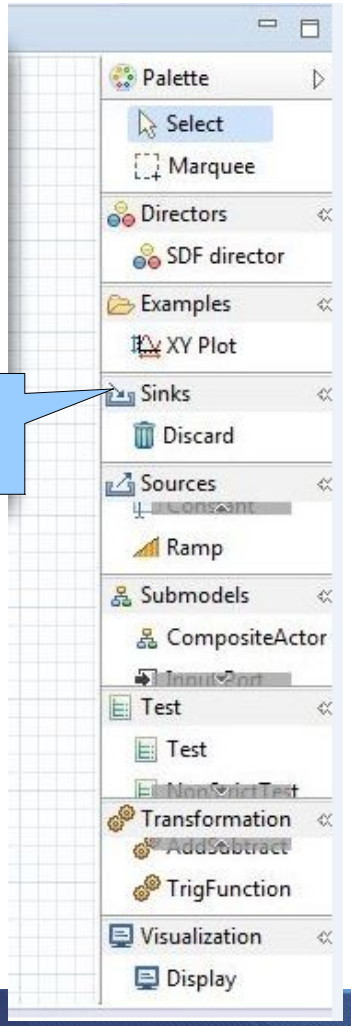
Eclipse frameworks used for the workflow editor :

- **Equinox**, Rich Client Platform (**RCP**),… : the traditional stuff for RCP apps.
- **Graphiti**: for the graphical workflow editor
- **Eclipse Modeling Framework** (EMF): to define a metamodel for Ptolemy II's model elements like Actors, CompositeActors, Parameters, Directors etc., for use by the Graphiti editor.
- **EMF Forms**: to define Actor configuration forms during the workflow design

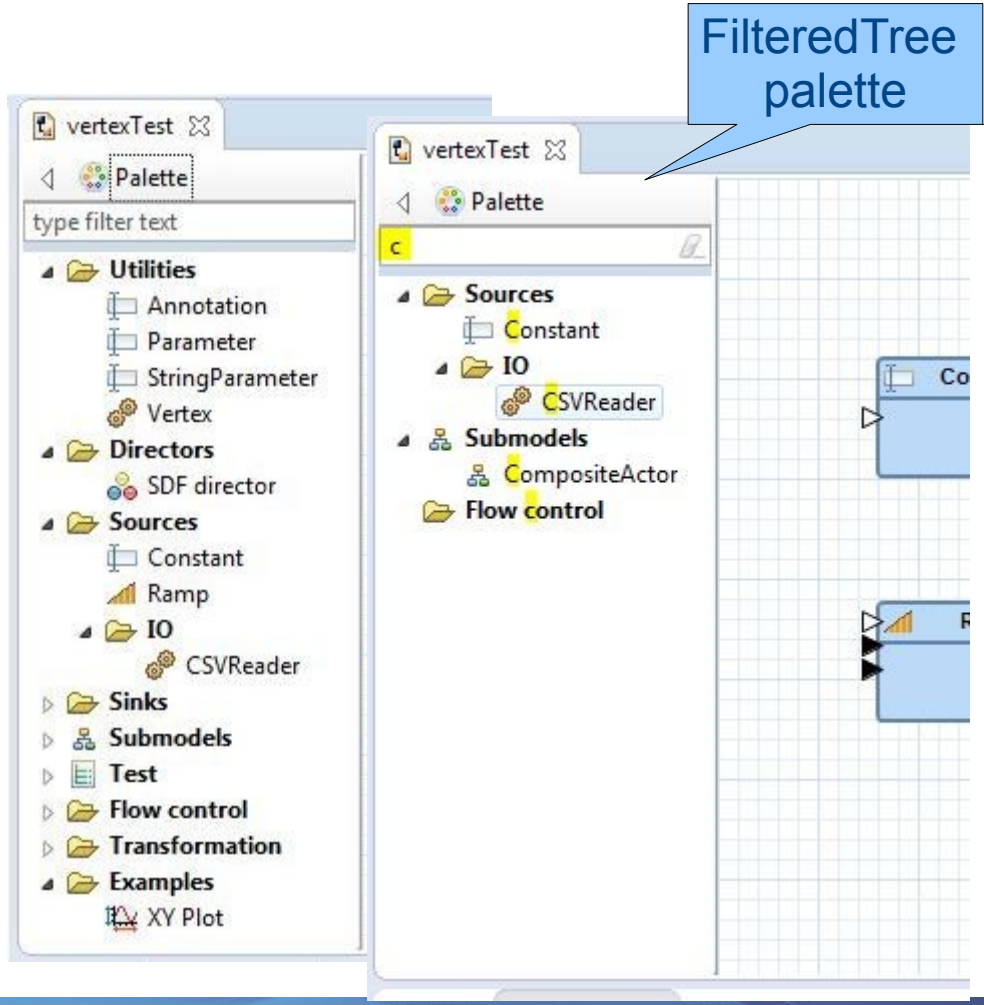


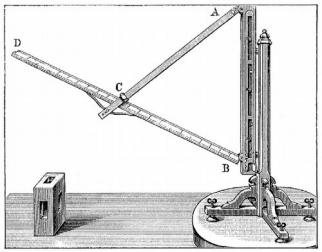
Graphiti : Palette tree

Classic palette



FilteredTree palette





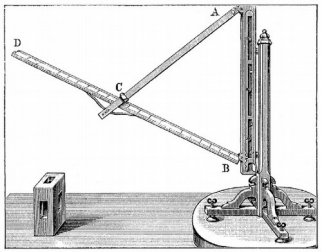
Workflow editor
status

Status :

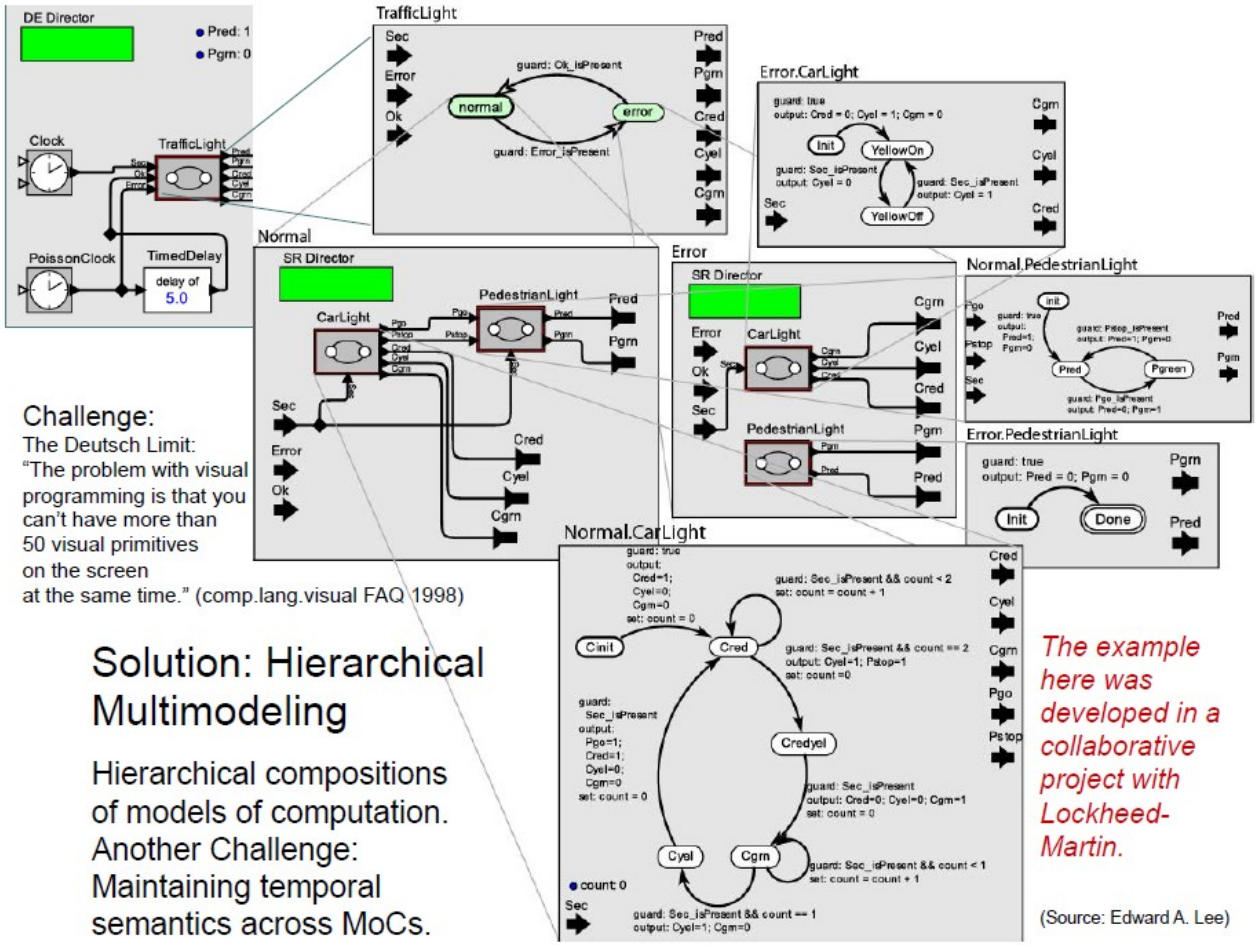
- Core underlying tools are integrated
- Single-level models
- Improved palette, configurable via extension points
- Custom shapes from SVG and Ptolemy xml

Next high priority :

- Support hierarchic Ptolemy II models
- Execution monitoring views



Hybrid hierarchical models



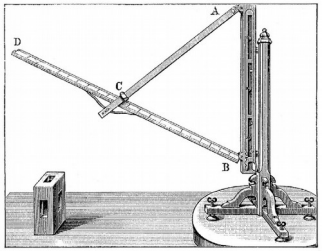
Challenge:
The Deutsch Limit:
"The problem with visual programming is that you can't have more than 50 visual primitives on the screen at the same time." (comp.lang.visual FAQ 1998)

Solution: Hierarchical Multimodeling

Hierarchical compositions of models of computation.
Another Challenge:
Maintaining temporal semantics across MoCs.

The example here was developed in a collaborative project with Lockheed-Martin.

(Source: Edward A. Lee)



Workflow runtime services

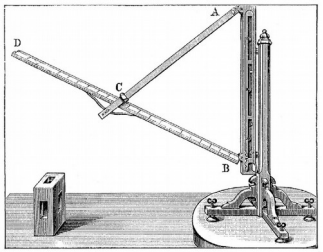
Maintaining models and running them

Goals :

- Usage in high-throughput workflow execution clusters
- Versioned model assets, simple activation & rollback of versions
- Run/debug locally or remotely

Approach :

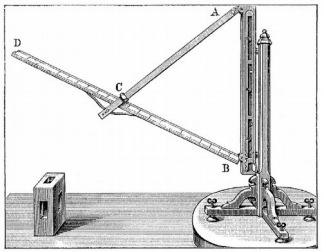
- Services `WorkflowRepositoryService & WorkflowExecutionService`
- Lightweight serializable `ModelHandle & ProcessHandle`
- Lazy loading of raw model definition and on-demand instantiation of live workflow elements



Workflow runtime services

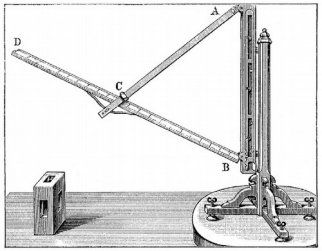
```
CompositeActor helloModel = ...;
ModelHandle handle = repositoryService.commit(HELLO_CODE, helloModel);
// some processing & do some updates in the model
// ...
ModelHandle updatedHandle = repositoryService.update(handle, updatedModel, false);
// some time later we can activate the updated model
// ...
repositoryService.activateModelRevision(updatedHandle);
// and some time later again someone would request the currently active model and get the new one
// ...
ModelHandle activeHandle = repositoryService.getActiveModel(HELLO_CODE);
```

```
ProcessHandle procHandle = processingService.start(StartMode.RUN, modelHandle, null, null, null);
// some time later we want to suspend the execution
ProcessHandle suspendHandle = processingService.suspend(procHandle);
// ...
assertEquals("Process should be SUSPENDED", ProcessingStatus.SUSPENDED, suspendHandle.getExecutionStatus());
// and then let's resume again
ProcessHandle resumeHandle = processingService.resume(suspendHandle);
// ...
assertEquals("Process should be RESUMED", ProcessingStatus.ACTIVE, resumeHandle.getExecutionStatus());
// we can also wait for the workflow to finish, with a timeout
processingService.waitUntilFinished(resumeHandle, 1, TimeUnit.SECONDS);
```



Agenda

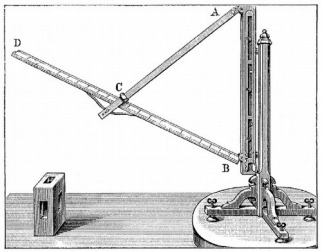
- Intro - Scientific software context
- Triquetrum overview
- Workflow features
- Task processing
- Integration approaches
- Project results
- Roadmap



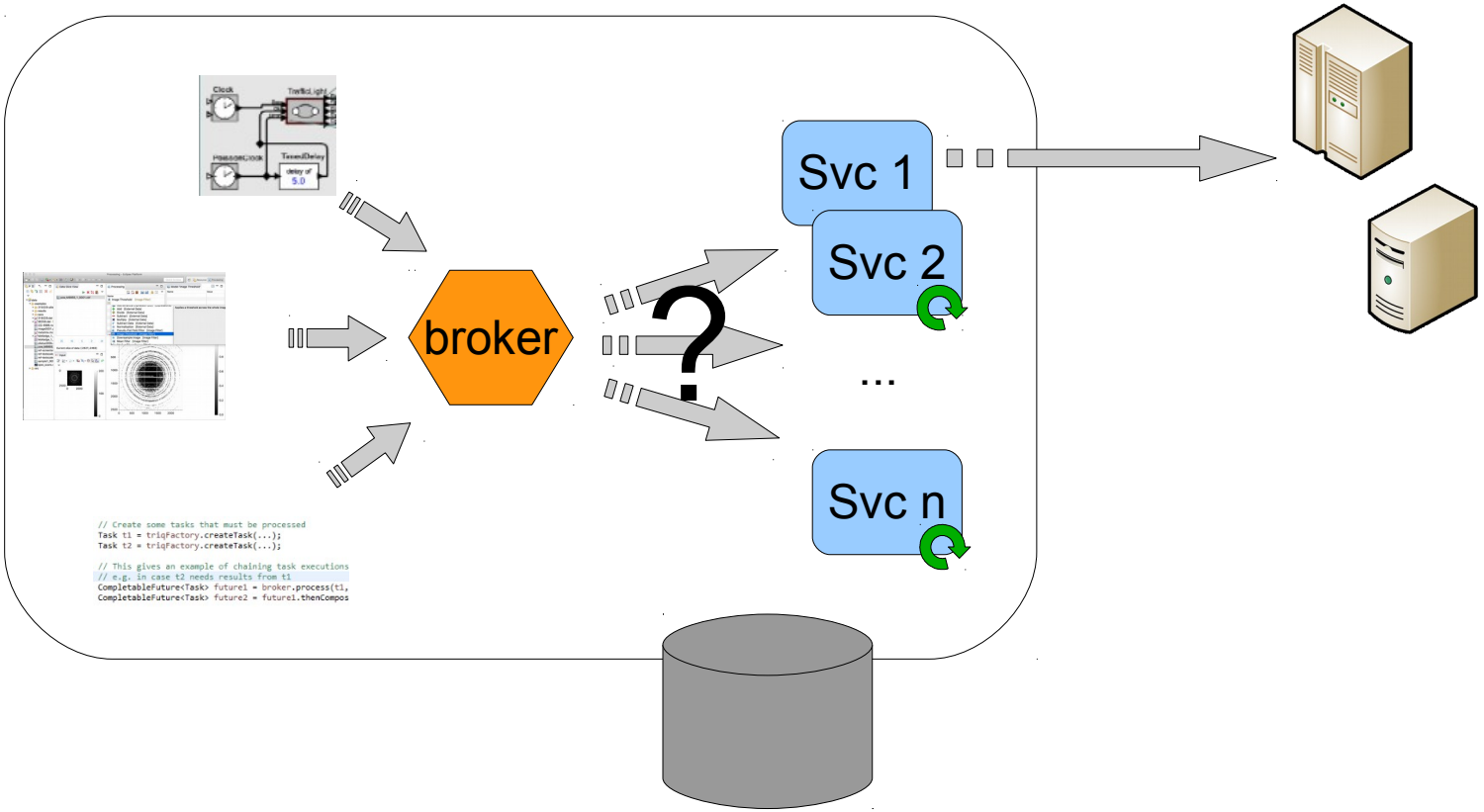
Workflows & Processes

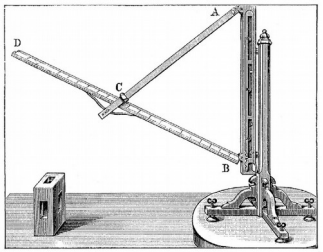
- A **Process** is performed as a **sequence of Tasks**
- A **Task** has
  - an **initiator** and **executor,** input **attributes** and (optional) **results**
  - a **life-cycle with start, finish, error, … events**
- Tasks get executed in **TaskProcessingServices**

- A Process can be driven from :
  - a predefined model, e.g. a Triquetrum/Ptolemy workflow model (actors can be task initiators)
  - ad-hoc user actions through a Task-based UI
- Execution traces and provenance info are automatically stored based on Tasks, events, results, errors



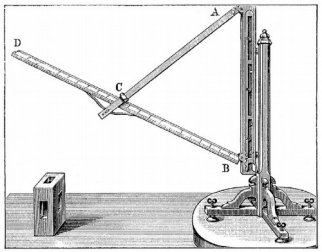
Getting a task done



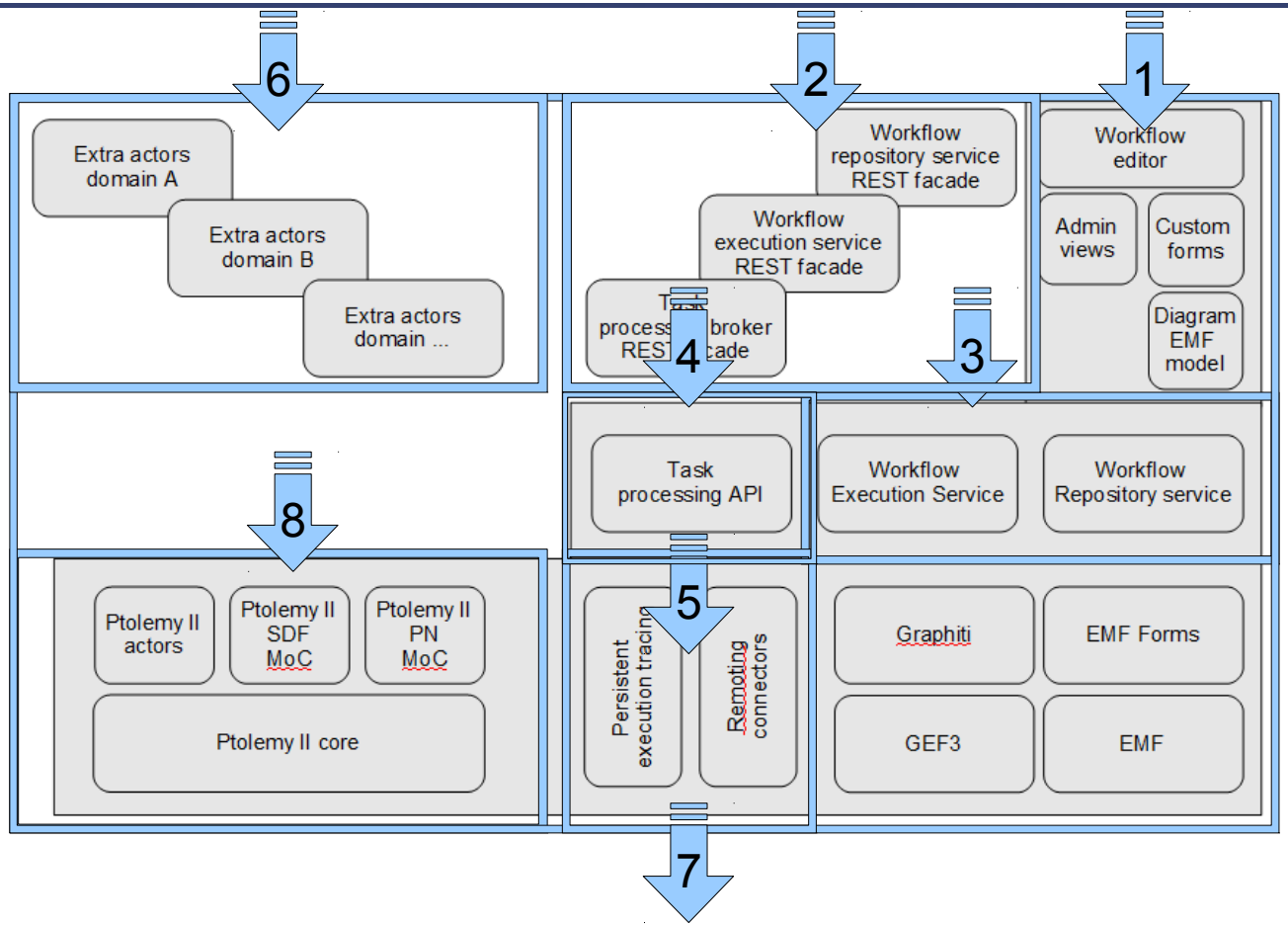


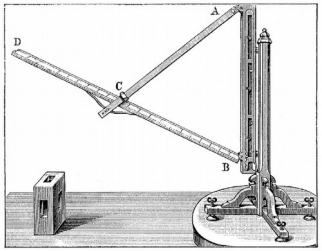
Agenda

- Intro - Scientific software context
- Triquetrum overview
- Workflow features
- Task processing
- Integration approaches
- Project results
- Roadmap



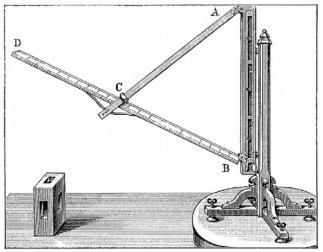
Integrating Triquetrum





Integrating Triquetrum

1. Workflows all the way
2. Embedded workflows – remote API
3. Embedded workflows – local API
4. Task submissions
5. Task processing service implementations
6. Domain-specific workflow actors
7. Triquetrum connectors
   (web-services and other protocols)
8. Ptolemy II OSGi bundles



INT-1 : Workflows all the way

Characteristics

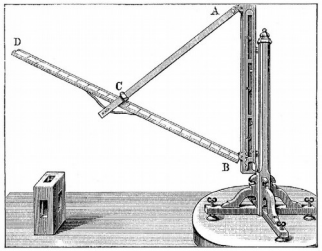
- Workflow models & diagrams exposed as real assets
- Users with different roles collaborate on design and maintenance

Integration approach

- Integrate Triquetrum in your RCP
- Launch workflows from inside the editor, or...
- Build own custom views from where to load and run workflows, using the corresponding Triquetrum services.

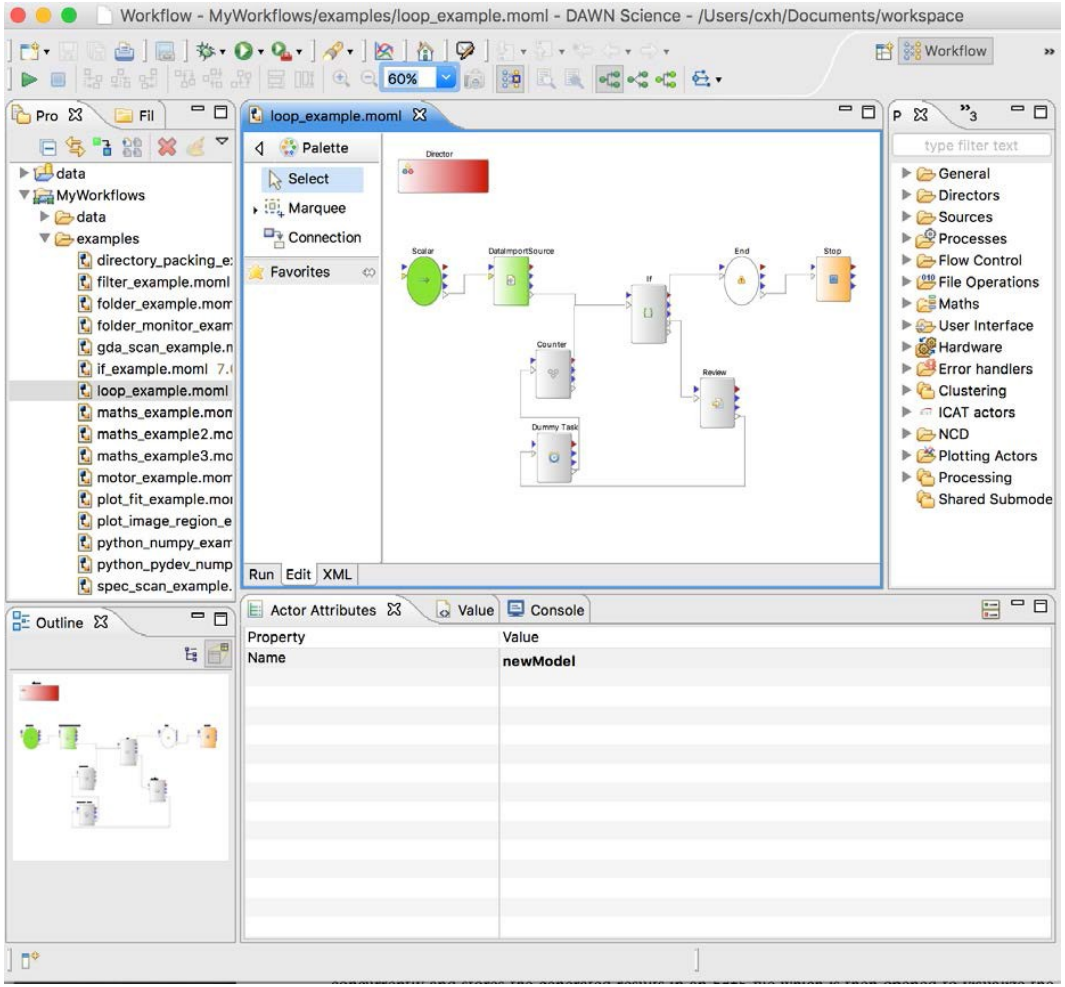
Examples (from Passerelle, predecessor of Triquetrum)

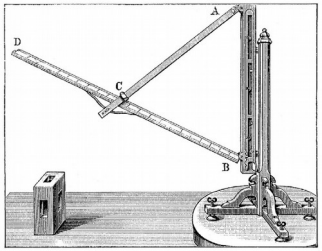
- Diamond Light Source : DAWN scientific workbench



DAWN and workflows

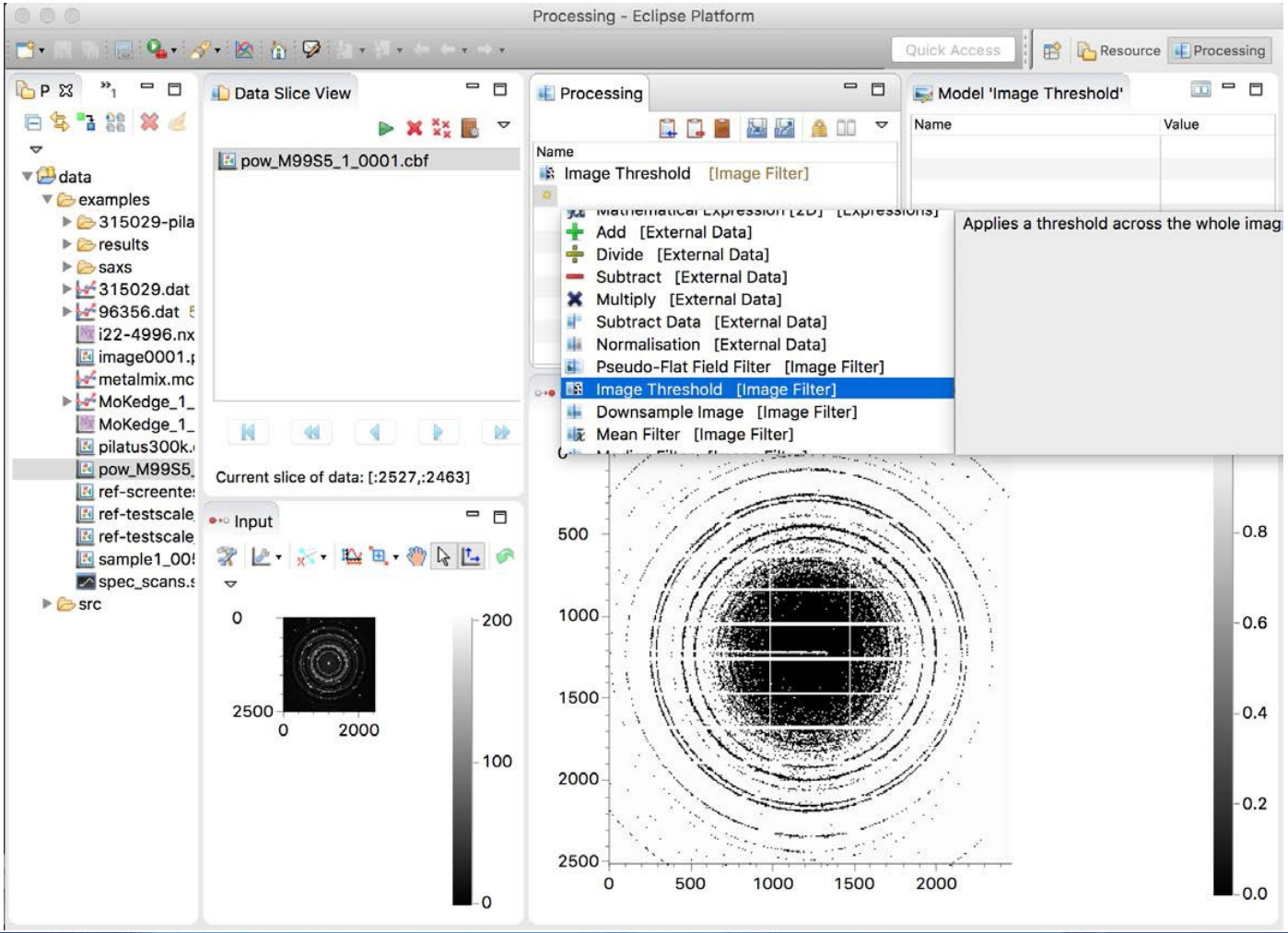
Dawn
Rise of the collaborative UI

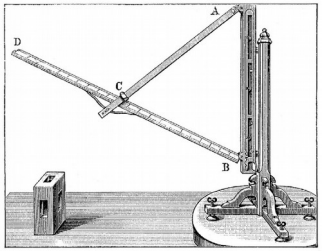




DAWN and workflows

Dawn
Rise of the collaborative UI

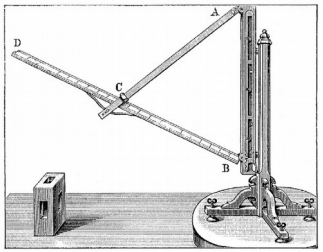




INT-2&3 : Embedded workflows
(invisible to user)

Characteristics

- Graphical models, even whole concept of workflows can remain invisible to the end-user
- Technical design decision to use Ptolemy II models internally e.g. to take advantage of :
  - actor/component-oriented assemblies with deterministic MoC
  - ease of maintenance, versioning
  - technical services like integrated non-blocking concurrency, error handling, execution traces, ...



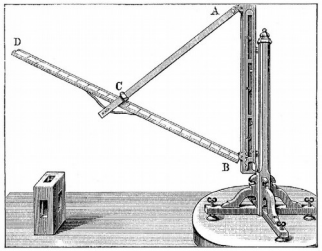
Embedded workflows
(invisible to user)

Integration approach

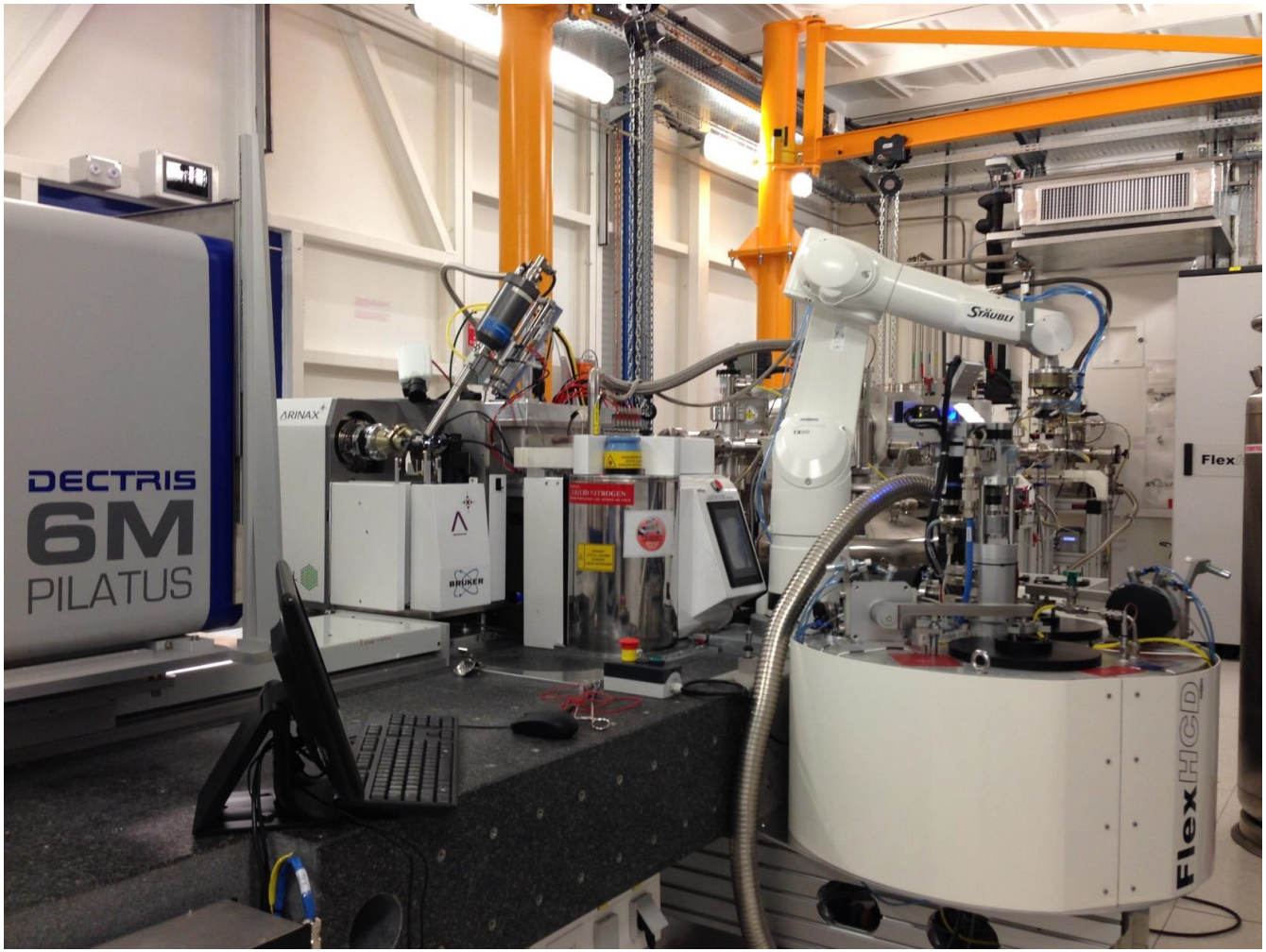
- Local integration : Include Triquetrum repo and execution services in your OSGi application. (and store the models in the repository somehow)
- Remote integration : e.g. from your Python application via REST access to a remote Triquetrum runtime

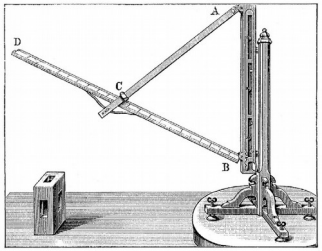
Examples

- (local) ICE Triquetrum integration POC
- (local) Synchrotron Soleil control widgets for workflows ; auto-generated control HMI (with Passerelle, Triquetrum's predecessor)
- (remote) ESRF & EMBL MASSIF beamlines use MxCube control GUI to execute workflows on cluster that drive automated experiments. (Passerelle)

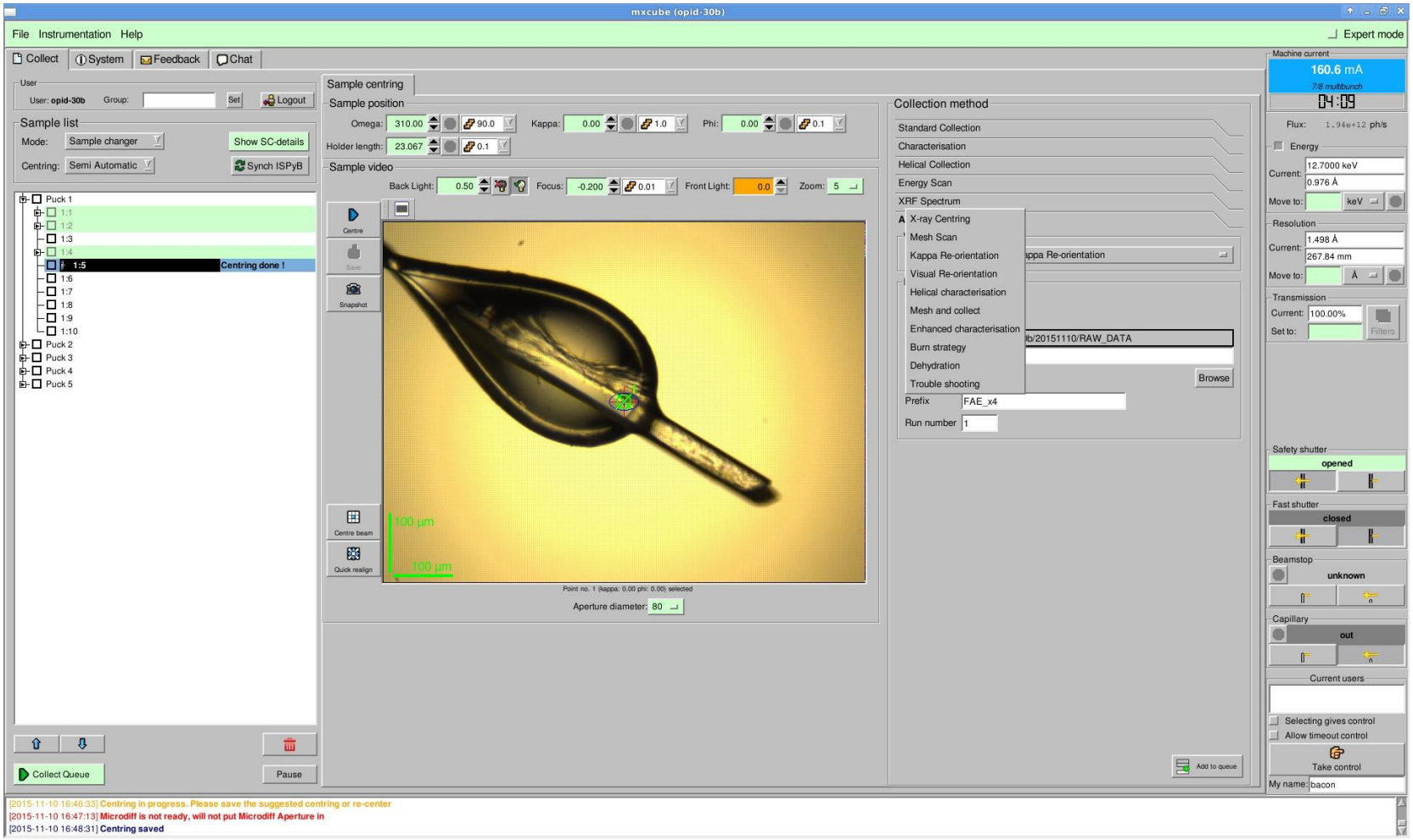


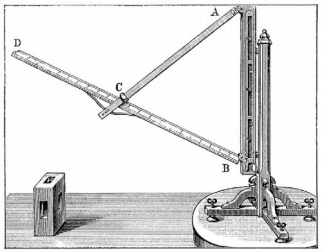
ESRF EMBL MASSIF





MxCube control software





ISPyB LIMS with final results

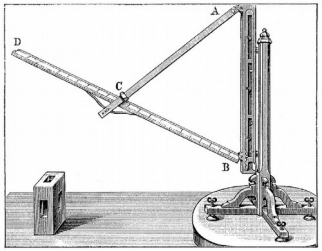
Save

Session

Expand All Collapse All Clear Grouping

Experiment Type	Image Prefix	Run#	Start time	Parameters	Comments	Image Thumbnail	Crystal snapshot	Graph	Second Graph	Results
MXPressE	opid291	1	10:13:37 30-04-2014	Nb tot images: 1402 Nb images: 1402 Exp. time: 0.04 s Phi range: 0.10 ° Flux: 2.74E11 photons/sec Detector resolution: 2.18 Å Transmission: 6.94 Wavelength: 0.976 Å Total expo time: 45.88 s						MXPressE EDNA dp GrenADES fp GrenADES pp Space Group: P 21 21 21 Completeness:
MXPressE	opid291	1	10:07:02 30-04-2014	Nb tot images: 1062 Nb images: 1062 Exp. time: 0.04 s Phi range: 0.20 ° Flux: 3.32E12 photons/sec Detector resolution: 2.00 Å Transmission: 100.00 Wavelength: 0.976 Å Total expo time: 33.30 s						MXPressE EDNA dp GrenADES fp GrenADES pp Space Group: P 21 21 21 Completeness:
MXPressE	opid291	1	10:00:32 30-04-2014	Nb tot images: 1062 Nb images: 1062 Exp. time: 0.04 s Flux: 3.35E12 photons/sec Detector resolution: 2.00 Å Transmission: 100.00 Wavelength: 0.976 Å Total expo time: 33.30 s						MXPressE EDNA dp GrenADES fp GrenADES pp Completeness:
MXPressE	opid291	1	09:53:43 30-04-2014	Nb tot images: 1282 Nb images: 1282 Exp. time: 0.04 s Phi range: 0.10 ° Flux: 2.85E11 photons/sec Detector resolution: 2.27 Å Transmission: 6.94 Wavelength: 0.976 Å Total expo time: 41.44 s						MXPressE EDNA dp GrenADES fp GrenADES pp Space Group: P 21 21 21 Completeness:
MXPressE	opid291	1	09:45:54 30-04-2014	Nb tot images: 1502 Nb images: 1502 Exp. time: 0.04 s Phi range: 0.10 ° Flux: 1.36E12 photons/sec Detector resolution: 2.44 Å Transmission: 38.26 Wavelength: 0.976 Å Total expo time: 49.58 s						MXPressE EDNA dp GrenADES fp GrenADES pp Space Group: P 21 21 2 Completeness:
Characterization	ref-p38_EDNA	1	09:38:21 30-04-2014	Nb images: 2 Exp. time: 0.04 s Phi range: 1.00 ° Flux: 5.47E11 photons/sec Detector resolution: 2.00 Å Transmission: 14.64 Wavelength: 0.976 Å Total expo time: 0.07 s						Indexing Strategy Space Group: Unit Cell a, b, c: alpha, beta, gamma: Mosaicity: Ranking Resolution:

ispyvalid.esrf.fr:8080/ispyb/user/viewResults.do?reqCode=viewJpegImageFromFile&file=/data/pyarch/id29/opid291/20140429/PROCESSED_DATA/1-6/MXPressE_01/Workflow_20140430-094137/html/Mesh_01/grid.png



Everything traced on workflow server

Passerelle EDM - Mozilla Firefox (on ub1404)

Passerelle EDM

mxedna:8080/BES/bridge/app/

Search

Beamline Expert System

ESRF | EMBL

Connected as: Olof Svensson

Version: 1.5.6

Requests

EN

Expert
- Flows
- Flow History
- Flow Feedback

Repository
- Preferences
- Project
- Flows
  - Translations
  - Flows
  - Shared Submodels
  - Flow Group
- Knowledge
  - Rule Packages
  - Rule files
- Media
  - Images
- Test
- Alert

Analysis
- Requests
- Tasks
- Results
- Events

Reports

Enter search string here | Search | Filter | My settings

Reference: | Id: | Correlation Id:

Type: | Initiator:

Status: Created | Error | Finished | Cancelled | Started | Timeout

Creation Date: yyyy-MM-dd HH:mm:ss To yyyy-MM-dd HH:mm:ss

View | Export | 1-100 of 31802 rows | 100 | Add Columns

Reference	Id	Type	Creation Date	Initiator	External Reference	Status	Duration(ms)
2903	275845	CreateThumbnails	2015-04-12 12:08:51	id29	mx1680	Finished	60,000
2903	275838	Characterisation	2015-04-12 12:08:27	id23eh1	mx1680	Finished	33,000
2903	275834	EDNA_dp	2015-04-12 12:08:18	id23eh1	mx1680	Finished	6,000
2903	275841	CreateThumbnails	2015-04-12 12:0				
2893	275817	MXPressE	2015-04-12 12:0				
2903	275826	EDNA_dp	2015-04-12 12:0				
2903	275810	Characterisation	2015-04-12 12:0				
2903	275806	EDNA_dp	2015-04-12 12:0				
2903	275767	CreateThumbnails	2015-04-12 12:0				
2903	275822	Characterisation	2015-04-12 12:0				
2903	275763	CreateThumbnails	2015-04-12 12:0				
2903	275688	EDNA_dp	2015-04-12 12:0				
2903	275802	Characterisation	2015-04-12 12:0				

Value

true

ImageQualityIndicators: ref-MOA-X19-A6-10-4_2_0001.cbf: good bragg 520, r1 2.9 [A], max cell 185.2
92.2
ImageQualityIndicators: ref-MOA-X19-A6-10-4_2_0002.cbf: good bragg 23, r1 3.9 [A], max cell 235.6 [
17.0
Indexing: laue/space group P3, mosaicity 2.20 [degree], RMS dev pos 0.36 [mm] ang 0.52 [degree]
Indexing: refined Cell: 121.07 121.07 99.28 90.00 90.00 120.00
Integration: 1 no full: 0, part: 14517, bad/neg/ovrlp: 7244, RMS dev: 0.138 [mm], I/sigma overall 0.5 a
Integration: 2 no full: 0, part: 15056, bad/neg/ovrlp: 8417, RMS dev: 0.130 [mm], I/sigma overall 0.1 a
Strategy: Radiation damage estimated, time to reach Henderson limit: 31.4 s
Strategy: Options: aimed completeness = 1.0, aimed resolution = 0.50 A, complexity = none
Strategy: Collection plan 1: Resolution limit is set by the radiation damage
Strategy: Ranking resolution: 2.58 [A]
Strategy: wedge 1: resolution 2.58 [A], sub wedge 1: start 108.00, images 600, width 0.10, time 0.04 [s
Strategy: total no images 600, total exposure time 22.2 [s]

Passerelle EDM - Mozilla Firefox (on ub1404)

Passerelle EDM

mxedna:8080/BES/bridge/app/

Search

Beamline Expert System

ESRF | EMBL

Connected as: Olof Svensson

Version: 1.5.6

Flow MXPressE (6.0) on 2015-04-12 12:07:32

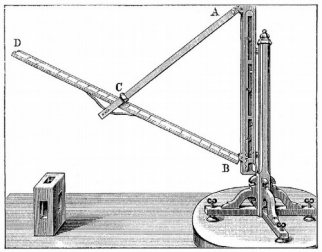
EN

Back | Edit | Validate | Flow | Run | Customize

MXPressE

Director | Error Handler by Severity | CommonErrorReporter | Start | Beamline setup | Prepare MXPressE | CommonPrepareExperiment | ExecuteMXPress | Set ISPyB to success | Workflow ended with error messages | Finished | Stop | Set ISPyB to success with error messages

iSencia

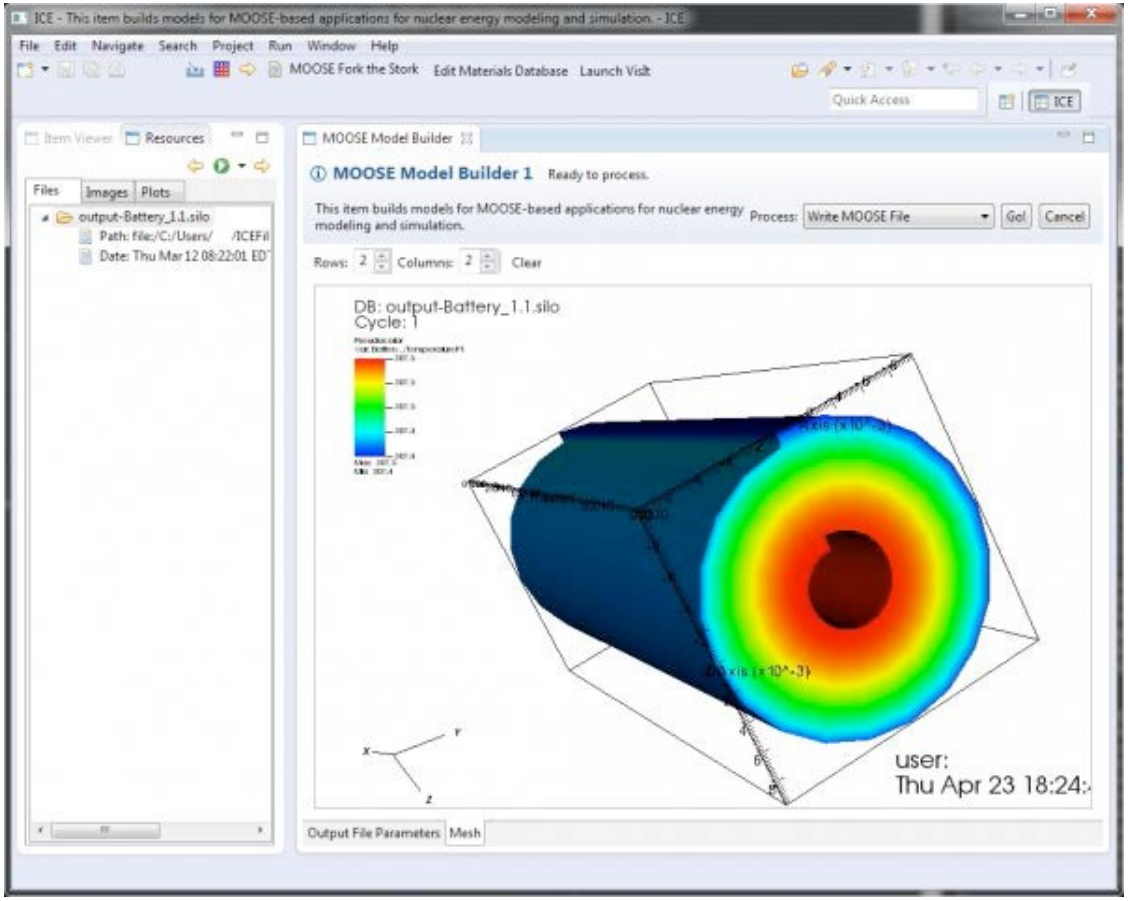


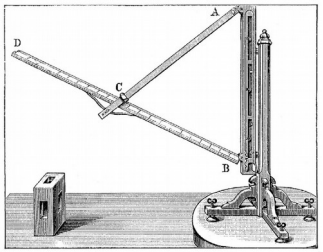
ICE

Scientific model setup, launching simulations, data analysis,...

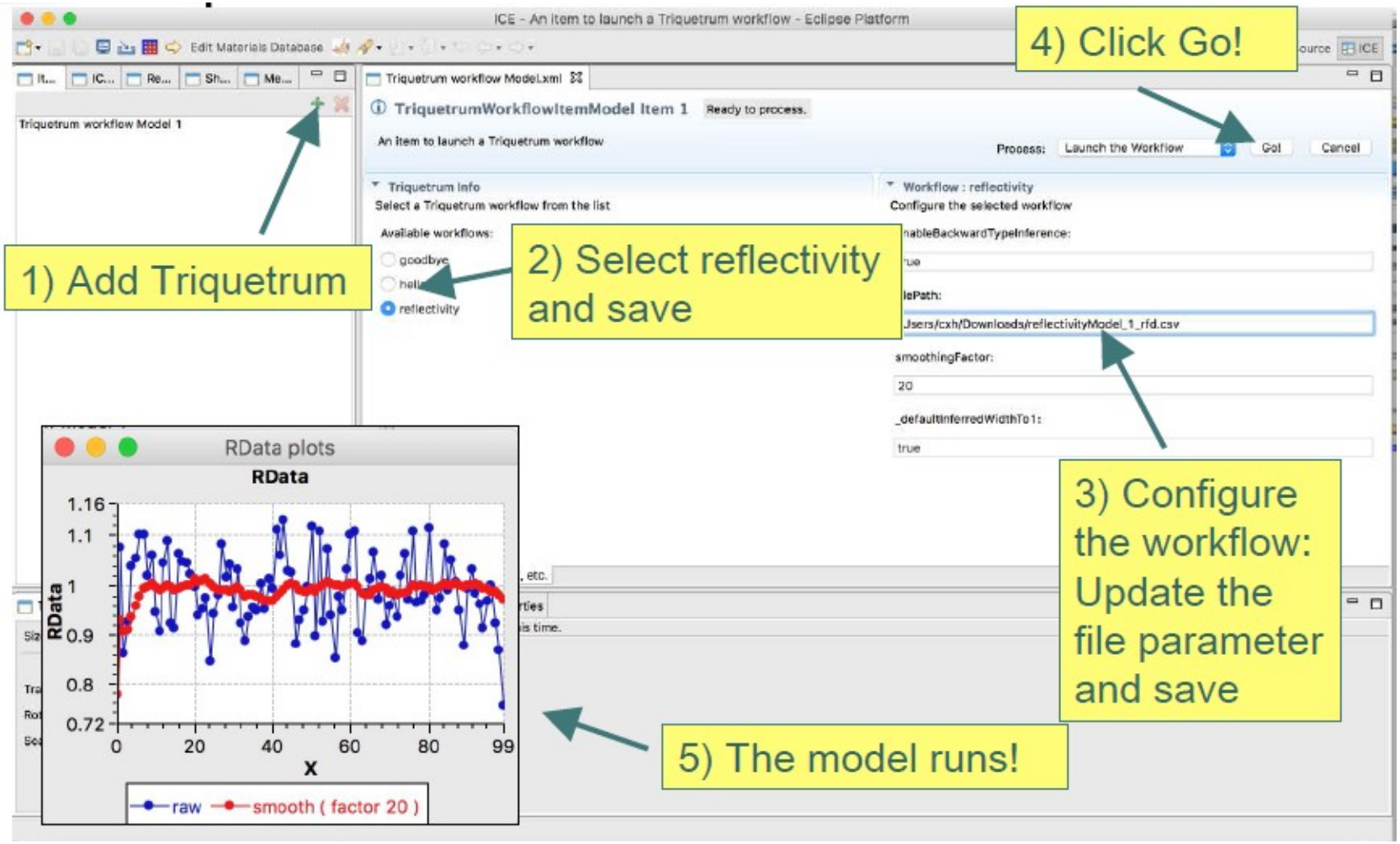
2009: start

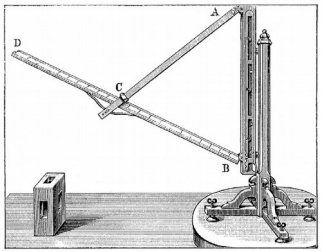
2014: Eclipse Science Project



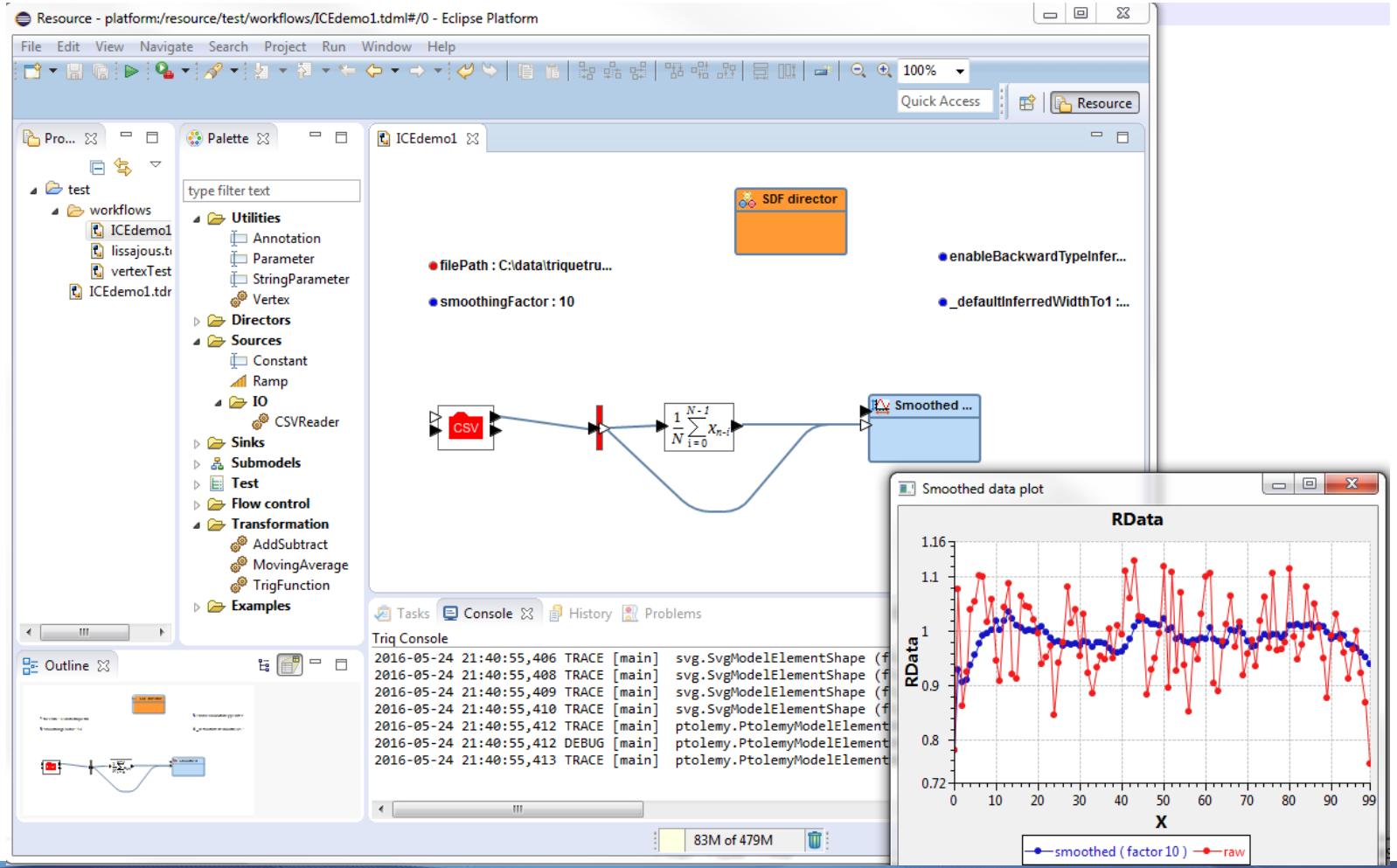


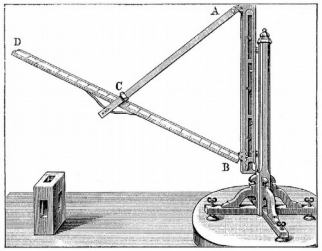
ICE – Triquetrum POC





Remember...

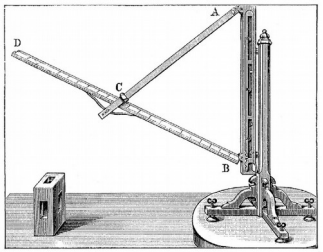

Resource - platform:/resource/test/workflows/ICEdemo1.tdml#/0 - Eclipse Platform
File Edit View Navigate Search Project Run Window Help
Quick Access
Resource
Palette
type filter text
Utilities
Annotation
Parameter
StringParameter
Vertex
Directors
Sources
Constant
Ramp
IO
CSVReader
Sinks
Submodels
Test
Flow control
Transformation
AddSubtract
MovingAverage
TrigFunction
Examples
ICEdemo1
SDF director
filePath : C:\data\triquetru...
smoothingFactor : 10
enableBackwardTypeInfer...
_defaultInferredWidthTo1 :...
CSV
Smoothed ...
Tasks
Console
History
Problems
Triq Console
2016-05-24 21:40:55,406 TRACE [main] svg.SvgModelElementShape (f
2016-05-24 21:40:55,408 TRACE [main] svg.SvgModelElementShape (f
2016-05-24 21:40:55,409 TRACE [main] svg.SvgModelElementShape (f
2016-05-24 21:40:55,410 TRACE [main] svg.SvgModelElementShape (f
2016-05-24 21:40:55,412 TRACE [main] ptolemy.PtolemyModelElement
2016-05-24 21:40:55,412 DEBUG [main] ptolemy.PtolemyModelElement
2016-05-24 21:40:55,413 TRACE [main] ptolemy.PtolemyModelElement
Outline
83M of 479M
Smoothed data plot
RData
RData
X
smoothed (factor 10)
raw



INT-5 : Task processing service implementation

Characteristics

- Allow Triquetrum tasks to be processed by your services
- Use Triquetrum's processing features (asynch API, error handling, …)
- Enable reuse and combination of different service families through a common coordination system (e.g. Triquetrum workflows)



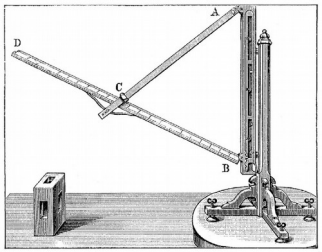
Task processing service implementation

Approach

- Decide on Task properties : Type identifier, Required & optional attributes
- Implement interface `o.e.t.processing.service.TaskProcessingService`
- Work with Task and its attributes as inputs, register progress as Events and (optionally) results as ResultBlocks & ResultItems.
- Register implementation as an OSGi service, e.g. using DS

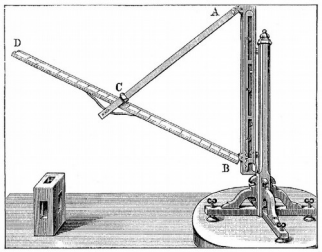
Examples

- Trivial example in `org.eclipse.triquetrum.processing.test`
- Your services?



Agenda

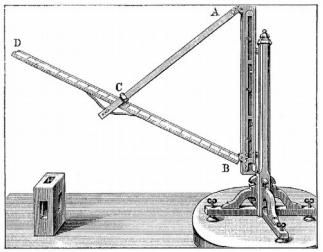
- Intro - Scientific software context
- Triquetrum overview
- Workflow features
- Task processing
- Integration approaches
- Project results
- Roadmap



The results

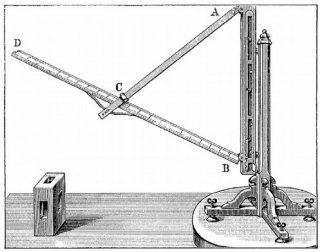
- The combination of Eclipse/OSGi with Ptolemy II delivers a solid platform for a wide range of workflow applications, especially scientific workflows.

- A powerful ecosystem for projects like Triquetrum comes from:
  - the modularity and dynamism offered by OSGi
  - the rich set of frameworks and technologies offered through the Eclipse Foundation,
  - and the community of the Eclipse Science Working Group



Roadmap

- First release : the Science 2016 release in October
  - Scope : Current status +
    - Support hierarchical models
    - Integration of Eclipse Layout Kernel
    - Using CDO as repository
    - Storing execution traces in RDB
- More integration cases
- Grow group of active committers



Project info

- Project site : https://projects.eclipse.org/projects/technology.triquetrum

- Sources : https://github.com/eclipse/triquetrum
- Wiki : https://wiki.eclipse.org/Triquetrum
- Blog : http://eclipse.github.io/triquetrum/
- Mailing list : https://dev.eclipse.org/mailman/listinfo/triquetrum-dev
- Hudson : https://hudson.eclipse.org/triquetrum/

iSencia BELGIUM

passerelle

SCIENCE eclipse.org

Evaluate the Sessions

eclipsecon
France 2016

Sign in and vote at **eclipsecon.org**

-1 0 +1