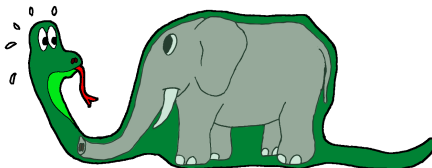


Pyphant

Klaus Zimmermann Lorenz Quack Andreas W. Liehr
`servicegruppe.wissinfo@fmf.uni-freiburg.de`

Servicegroup Scientific Data Processing
Freiburg Materials Research Center
University of Freiburg

Europython2006.tex 1029 2006-06-25 21:35:51Z zklaus



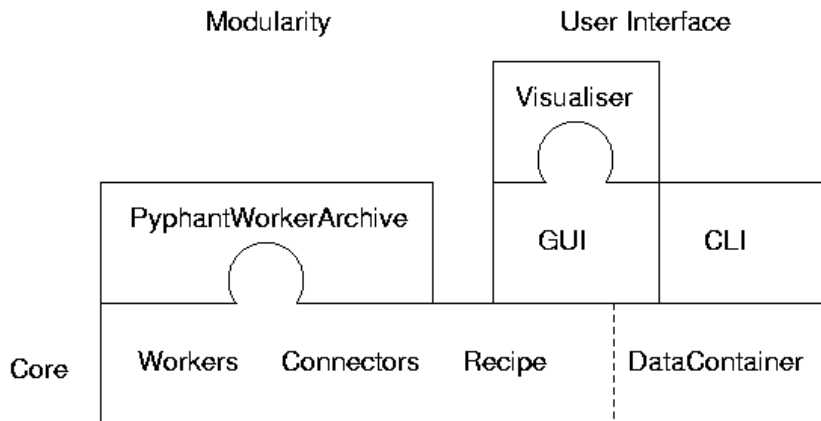
- 1 Introduction
- 2 The Core
- 3 The User Interfaces
- 4 Summary and Future

Concepts

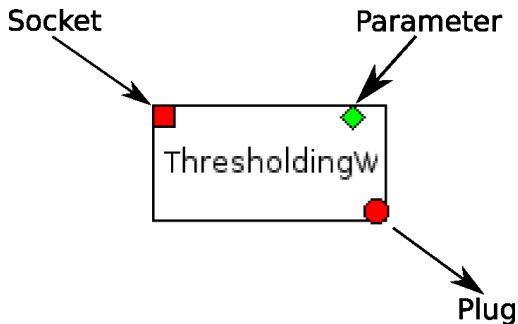
Pyphant is a framework for the construction of reusable data processing and analysis workflows that is

- layered
- multi-threading capable
- extensible

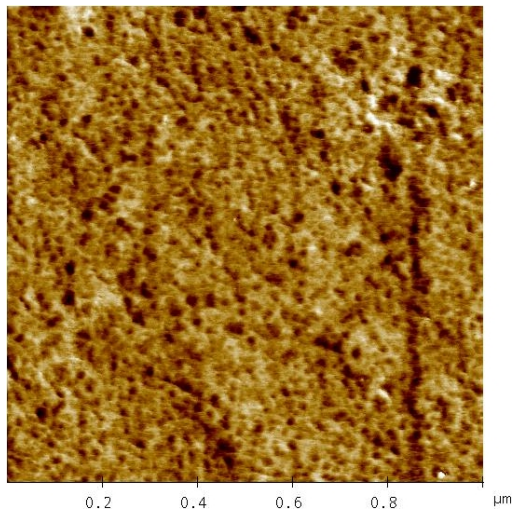
Structure



The Worker



The Problem

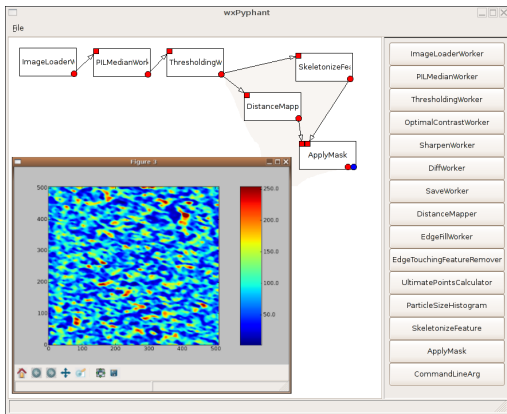


Extract a particle size distribution from a microscopic image.

The Pyphant Approach

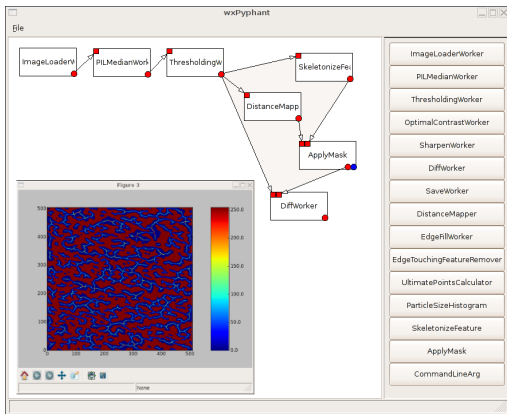
- Visual Programming
- Immediate Feedback
- Incremental Adaption

The Pyphant Approach



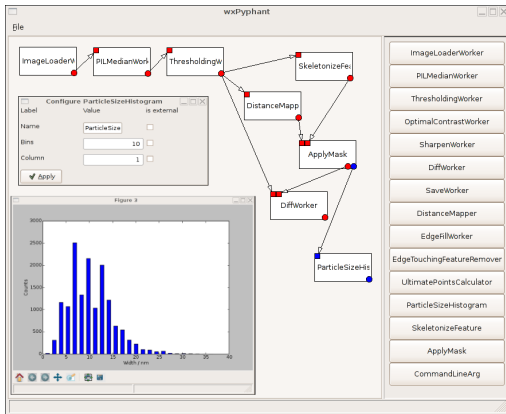
- Visual Programming
- Immediate Feedback
- Incremental Adaption

The Pyphant Approach



- Visual Programming
- Immediate Feedback
- Incremental Adaption

The Pyphant Approach



- Visual Programming
- Immediate Feedback
- Incremental Adaption

DiffWorker - the Worker Explained

```
6 def createWorker(recipe , annotations={}):
7     return DiffWorker(recipe , annotations)
8
9 WORKER.INFO = Worker.WorkerInfo(" DiffWorker" ,
    createWorker)
10
11 class DiffWorker(Worker.Worker):
12     _sockets = [ ("image1" , Connectors.TYPE_IMAGE) ,
13                 ("image2" , Connectors.TYPE_IMAGE) ]
14
15     @Worker.plug(Connectors.TYPE_IMAGE)
16     def diffImages(self , image1 , image2):
17         im1=image1.getSliceAsImage()
18         im2=image2.getSliceAsImage()
19         result=ImageChops.difference(im1 , im2)
20         return DataContainer.DataContainer(result)
```

The Connectors

Connectors

Connectors model the edges of the computing graph.

This graph is called a *Recipe*. Three kinds of connectors:

- Sockets
- Parameters
- Plugs

Sockets and Parameters

Sockets form the input facilities of the workers.

One socket plays host to at most one *Plug*.

If a socket is queried by its worker, it relays that query to that plug.

A parameter is a special socket.

The Connectors

Connectors

Connectors model the edges of the computing graph.

This graph is called a *Recipe*. Three kinds of connectors:

- Sockets
- Parameters
- Plugs

Sockets and Parameters

Sockets form the input facilities of the workers.

One socket plays host to at most one *Plug*.

If a socket is queried by its worker, it relays that query to that plug.

A parameter is a special socket.

The Plugs

Plugs

- are the computation entities.
- receive their input from the sockets of their workers.
- cache their results in a threadsafe way.

Execution of One Plug

Plugs are wrapped in automatically generated wrappers.
Those query the sockets asynchronously and call the plugs like ordinary methods.

The Plugs

Plugs

- are the computation entities.
- receive their input from the sockets of their workers.
- cache their results in a threadsafe way.

Execution of One Plug

Plugs are wrapped in automatically generated wrappers. Those query the sockets asynchronously and call the plugs like ordinary methods.

The Evaluation Model of the Entire Recipe

- Recipes are *not* executed, but evaluated *lazily*.
- Branched recipes lead to automatic multi-threading.
- Locking is taken care of by Pyphant.

The DataContainer

The DataContainer is intended as a common data interchange container that

- facilitates interoperability.
- provides infrastructure for dealing with unit affected data.
- is based on scipy arrays.
- supports images, possibly other formats.

The User Interfaces

wxPyphant - the Graphical User Interface

- Allows easy construction of recipes

The Command Line Interface

- Enables batch application of pre-defined recipes to sets of data.
- Does *not* depend on graphical environment.
- Is thus well suited to environments with great computing powers, but poor interfaces, like GRIDs.

The User Interfaces

wxPyphant - the Graphical User Interface

- Allows easy construction of recipes

The Command Line Interface

- Enables batch application of pre-defined recipes to sets of data.
- Does *not* depend on graphical environment.
- Is thus well suited to environments with great computing powers, but poor interfaces, like GRIDs.

Summary

Summary

Pyphant offers

- reusability
- extensibility
- data type independence
- separation of GUI and model, commandline applications
- multi-threading

Future

Future

In the future we plan to extend by Pyphant by

- More workers: Ill-posed problems, more imageprocessing, ...
- Better commandline support.
- Cyclic recipes, thus introducing controlling loops, iterative procedures, ...

The End

Acknowledgement

Thanks to Michael Röttger for the Pyphant logo and fruitful discussions.

