OpenAlea: Scientific Workflows
Combining Data Analysis and Simulation

Christophe Pradal
Christian Fournier
Patrick Valduriez
Sarah Cohen-Boulakia
The size of the data sets has increased
  ◦ Next Generation Sequencing, phenotypic data

The complexity of scientific analysis has increased
  ◦ From queries to complex integration pipelines

Scientific workflow systems provide solutions
  ◦ GUI drag&drop tools to be combined
  ◦ Provenance modules to keep track of data used/produced during an execution
  ◦ Scheduling execution in clouds/grids...

However, such systems failed to easily couple analysis and simulation
  ◦ Retro-action needed in system biology, developmental biology, ecology...
Outline

- Context of the work
- Specific use case and associated needs
- The OpenAlea workflow system
- Implementation of the use case
- OpenAlea community
- Conclusions
Phenoarch platform

- Study of the impact of the environment on the plants
- 1,600 plants
- Automated high throughput system
  - Imaging (side and top view images)
    - 52 GB/day
    - 2.75 TB /essay (50 days exp.)
    - 11 TB/year
  - Watering and whole-plant transpiration
    - Meristem temperature + weight is measured every 15min in control plants
Overview of Phenoarch

Montpellier, FRANCE

Conveyor belts

Watering stations

Sarah Cohen-Boulakia, Open Alea, SSDBM 2015, June 30th 2015
Result of 8 experiments: Each performed on 1600 plants
One use case

- Plants genotypes
  - DNA sequences
  - Allele sets
- Plants phenotypes
  - Size of the plants
  - Length of the leaves
  - Number of tillers (stems)...

→ Correlation between genotypes/phenotypes?

→ Impact of the environment?
  - Light
  - Nutriments

→ Simulation needed

- Plant growth impacted by the amount of light intercepted by the plant
- The light intercepted by the plant depends on the plant growth
  → Retro-action
Outline

- Context of the work
- Specific use case and associated needs
- The OpenAlea workflow system
- Implementation of the use case
- OpenAlea community
- Ongoing work & Conclusions
OpenAlea design principles

- **Scientific workflow system**
- **Language centric**
  - Common modeling language
  - Glue language
- **Component architecture**
  - Dynamic composition
  - High-level dataflow approach
- **Visual programming**
  - Graphical model representation
  - Automatic GUI generation
Visual Programming

Dataflow

Widgets

Component

Package Manager

Python Interpreter

Sarah Cohen-Boulakia, Open Alea, SSDBM 2015, June 30th 2015
Component-based architecture

- Simple Python function
  - Multiples I/O (ports)
  - Typed interface
  - Documentation

**Port**
- name: `nb_plants`
- Interface: `IInt`

```python
def regular(nb_plant, nb_rank, dx, dy):
    ""
    Calculates a regular plant distribution
    nb_plants : total number of plants
    nb_rank : total number of rows
    dx, dy : distance between 2 plants on the x-axis and the y-axis
    Return a list of (x,y) position
    ""
    nx = int(nb_plant / nb_rank)
    ny = nb_rank

    return [(i*dx, j*dy)
            for j in range(nb_rank)
            for i in range(nx)]
```

```
Name: regular distribution
Package: __my package__
Documentation:

Calculates a regular plant distribution
nb_plants : total number of plants
nb_rank : total number of rows
dx, dy : distance between 2 plants on the x-axis and the y-axis
Return a list of (x,y) position
```
OpenAlea - Concepts

- Workflow – Actors
  - \{X, Y, f\}

- Dataflow Variable
  - Transforms a workflow into a lambda-function

- Algebraic Operators
  - map, reduce, filter...
OpenAlea – Execution

- **Model-driven**
  - execution launched in response to requests for data in one actor (output port)
  - upstream subworkflow (connected to input ports) to be (recursively) executed
  - When an actor has received all data on input ports, it executes and places data on its output ports.

- **Blocked actors**
  - execution is not propagated

- **Lazy actors** (by default)
  - execution (re)performed only if new inputs
Reproducibility in OpenAlea

- **Provenance**
  - Logging during execution
  - PROV-DM

- **Workflow executions saved into notebooks**
  - Actors of the workflow → cells in the notebook
  - Data produced and used (execution) can be visualized

```python
In [1]: sum([_ for _ in range(N)])
```

**IP[y]:** *IPython Interactive Computing*

- Web-based interactive computational environment
- Combination of code execution, text, mathematics, plots and rich media into a single document
Outline

- Context of the work
- Specific use case and associated needs
- The OpenAlea workflow system
- Implementation of the use case
- Open Alea Community
- Ongoing work & Conclusions
Implementation of the use case

- Execution starts when the user clicks on the output port of *plot*
Implementation of the use case

- Plant Generation generates one (virtual) plant
- Map generates n virtual plants with some variability
- While to be executed
  - Left branch = init.
  - reduce(map(plant generation, internal variability), ...)
  - Exp design: space between plants...
Implementation of the use case

- *reduce* populates the virtual crop (union of all the plants generated)
- The stop condition is the flowering
Implementation of the use case

- The loop implements the retroaction between plant growth and light interception
  - Meteo01.csv: environmental conditions in the crop
Implementation of the use case
Using subworkflows (virtual experiment)
**IPy: Notebook**

**Scientific workflows meet modeling and simulation**  Last Checkpoint: Mar 18 18:39 (autosaved)

---

**OpenAlea: Scientific workflows meet modeling and simulation**

```python
In [2]:
from vpltkdisplay import *
from IPython.display import display
from openalea.plantgl.all import *
from openalea.core import *
import numpy as np
```

**Simulation of the growth of a crop**

```python
In [3]:
pkname='alinea.adel.tutorials.ssdbm'
node='5- while'
display(Dataflow(pkname, node))
```
Outline

- Context of the work
- Specific use case and associated needs
- The OpenAlea workflow system
- Implementation of the use case
- Open Alea Community
- Ongoing work & Conclusions
Figures: 580,000 downloads, 20 active developers

Free Software License
  - OpenAlea: CeCILL-C (aka L-GPL)
  - Components / Models: Open or Close source license

Peer Review, Tests, Code Quality, Documentation...

Collaborative Development
  - Shared development between various teams
  - Shared Methodology and Best Practices

Coding Sprint: Pair programming and test driven development
How big is OpenAlea?

- Core workflow system (IT)
- Vplants: Libraries, visualization (stats)
- Alinea: models, inputs (bio)
- Total: 957K Source Lines of Code

Credits: David Wheelers SLOCcont
Many uses of OpenAlea

Plant architecture analysis - VPlants
(Godin, Guedon et al)

Biophysical models
Buissiere et al., 07
Fournier et al., 07
Pradal et al., 08

3D reconstruction, simulation of development, and epidemics
Wheat, Maize, Rice, Vine
Apple, Almond, Peach
Outline

- Context of the work
- Specific use case and associated needs
- The OpenAlea workflow system
- Implementation of the use case
- Open Alea Community
- Conclusions
Conclusion

- Unique solution to easily deal with data analysis, modeling and simulation (retro-action)

- OpenAlea is a scientific workflow system actively used by the plant community since 2007

- Ongoing work
  - OpenAlea on Grids (FranceGrille)
  - Automatic generation of iPython notebooks based on workflow executions (reproducible papers)
  - Virtualization techniques to capture working environments and reproduce experiments
Aknowledgements

Christophe Pradal

Christian Fournier

Patrick Valduriez

Pascal Neveu

Christophe Godin
OpenAlea 2.0 infrastructure

Store workflows and data

OpenAlea server

Package Manager
Find, discover Services & Data

Scheduler
Distributed computing

Cyber Plant Community

Find

Publish
Share

Design
Interact

VisuAlea Web browser
HTML5, WebGL
Liquid VM: Reproducibility in OpenAlea

In collaboration with Dennis Shasha