pycropml Documentation

Release 0.0.2

Cyrille Ahmed Midingoyi

Apr 10, 2018

Contents

CHAPTER 1

Contents

Summary

Version 0.0.2

Release 0.0.2

Date Apr 10, 2018

Author See **'authors'**_ section

ChangeLog See **'changelog'**_ section

1.1 What is PyCropML?

PyCropML is a free, open-source library for defining and exchanging CropML models.It is used to generate components of modeling and simulation platforms from the CropML specification and allow component exchange between different platform.

It allows to parse the models described in CropML format and automatically generate the equivalent executable Python, java, C#, C++ components and packages usable from existing crop simulation platform.

1.1.1 What is CropML ?

CropML is a language based on XML format that allows to represent different biological processes involved in the crop models.

CropML project is intented to provide a common framework for defining and exchanging descriptions of crop growth models between crop simulation frameworks.

1.1.2 Objectives

Our main objectives are:

- define a **declarative language** to describe either an atomic model or a composition of models
- add semantic dimension to CropML language by annotation of the models to allow the composition of components of different platforms by using the standards of the semantic web
- develop a library to allow the transformation and the exchange of CropML model between different Crop modelling and simulation platform
- provide a **web repository** enabling registration, search and discovery of CropML Models
- facilitate Agricultural Model Exchange Initiative

1.1.3 Motivation

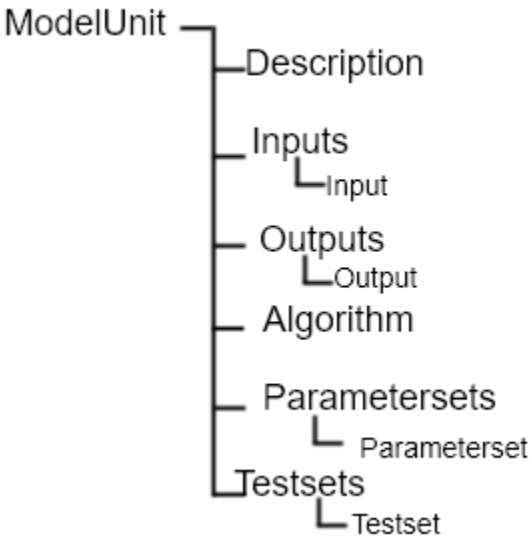
Nowadays, we observe the emergence of plant growth models which are built in differents platforms. Although standard platform development initiatives are emerged, there is a lack of transparency, reusability, and exchange code between platforms due to the high diversity of modeling languages leading to a lack of benchmarking between the different platforms.
This project aims to gather developers and plant growth modellers to define a standard framework based on the development of declarative language and libraries to improve exchange model components between platforms.

1.2 CropML Description

In CropML, a model is either a model unit or a composition of models unit. A ModelUnit represents the atomic unit of a crop model define by the modelers. A model composition is a model resulting from the composition of two or more atomic model.
These models have a specific formal definition in CropML.

1.2.1 Formal definition of a Model Unit in CropML

The structure of a Model Unit in CropML must be conform to a specific Document Type Definition named ModelUnit.dtd .
So a Model Unit CropML document is a XML document well-formed and also obeys the rules given in the ModelUnit schema.
This structure may be described by the below tree:



In the next, we define the major elements of a CropML model unit.

ModelUnit element

An atomic model in CropML is declared with <<ModelUnit>> element, the usual root of CropML ModelUnit document.

This element must contain a Description, an Algorithm, Parametersets and Testsets elements and may optionally have Inputs and Outputs elements. The restriction of the length of different lists is not imposed.

ModelUnit element must have an unique id and name attributes which are used to reference an atomic model. It also contains a timestep and unit attribute. The unit attribute is used to define the unit that are associated with the model.

Description element

This element gives the general information on the model and is composed by a set of character elements. It must contain Title, Authors, Institution and abstract elements and may optionally contain URI and Reference elements.

Inputs elements

The inputs of Model are listed inside an XML element called Inputs within a dictionary structure composed by their attributes which declarations are optional(default, max, min, parametertype, variabletype and URI) or required(name, datatype, description, inputtype, unit) and their corresponding value. *Inputs* element must contain one or more inputs elements.

The required *datatype* attribute is the type of input value specified in *default* (the default value in the input), *min* (the minimum value in the input) and *max* (the maximum value in the input). It may be one type of the set of types used in the existing crop modeling platform.

The *inputtype* attribute makes it possible to distinguish the variables and the parameters of the model. So it must take one of two possible values: *parameter* and *variable*.

The *parametertype* attribute defines the type of parameter which is specified by one of the following values: *constant*, *species*, *soil* and *genotypic*.

The *variabletype* defines the type of variable depending on whether it is a *state* or *rate* variable. State variable characterize the behavior of the model and rate variable characterizes the change of a state of the model.

Outputs element

The outputs of Model are listed inside an XML element called Outputs within a dictionary structure composed by their attributes which declarations are optional(variabletype and URI) or required(name, datatype, description, unit, max and min) and their corresponding value *Outputs* must contain zero or more output element. The definition of different attributes is same as Input's attributes.

Algorithm element

The *Algorithm* element defines the building block of CropML model unit and shows the computational method to determine the outputs from the inputs.
It consists of a set of mathematical equations (relation between inputs), loops and conditional instructions which are well structured in a specific *language*, the algorithm's attribute.

Parametersets element

Parametersets element contains one or more *Parameterset* elements that define the different ways of setting the model. Each *Parameterset* element must have *name* and *description* attributes that respectively represents the name and the description of each setting.
The different parameterset must contain a list of Param elements that show in attribute the name of the parameter (an input which inputtype equals *parameter*) and the fixed value of this one.

Testsets element

Testsets element contains one or more *Testset* elements that define the different run for evaluating the outputs of the model. Each *Testset* element must have *name*, *description* and *parameterset* attributes that respectively represents the name, the description of each run and the name of the parameterset related to the Testset. This one allow to retrieve the name and the value of different parameters includes in this parameterset.
The different Testset must contain a list of InputValue and OutputValue elements corresponding respectively to the values of inputs used in the run and the values of Outputs that will be asserted.

1.2.2 Formal definition of a Model Composition in CropML

The structure of a Model Unit in CropML must be conform to a specific Document Type Definition named Model-Composition.dtd .

1.3 PyCropML User Guide

Version 0.0.2

Release 0.0.2

Date Apr 10, 2018

This reference manual details functions, modules, and objects included in OpenAlea.Core, describing what they are and what they do. For a complete reference guide, see core_reference.

Warning: This "Reference Guide" is still very much in progress. Many aspects of OpenAlea.Core are not covered.

1.3.1 Manual

Note: The following examples assume you have installed the packages and setup your python path correctly.

Overview

Installation

```
conda install -c openalea pycropml
```

or

```
python setup.py install
```

Overview of the different classes

1.4 src

1.4.1 pycropml package

Submodules

pycropml.algorithm module

class pycropml.algorithm.**Algorithm**(*language*, *development*)
Bases: object

pycropml.checking module

class pycropml.checking.**Test**(*name*)
Bases: *pycropml.checking.Testset*

class pycropml.checking.**Testset**(*name*, *parameterset*, *description*, *uri=None*)
Bases: object

Test

pycropml.checking.**testset**(*model*, *name*, *kwds*)

pycropml.description module

class pycropml.description.**Description**
Bases: object

Model Unit Description.

A description is defined by:

- Title
- Author
- Institution
- Reference
- Abstract

pycropml.inout module

class pycropml.inout.**Input**(*kwds*)
Bases: *pycropml.inout.InputOutput*

class pycropml.inout.**InputOutput**(*kwds*)
Bases: object

class pycropml.inout.**Output**(*kwds*)
Bases: *pycropml.inout.InputOutput*

pycropml.modelunit module

class pycropml.modelunit.**ModelDefinition**(*kwds*)
Bases: object

class pycropml.modelunit.**ModelUnit**(*kwds*)
Bases: pycropml.modelunit.ModelDefinition

Formal description of a Model Unit.

add_description(*description*)
TODO

pycropml.parameterset module

class pycropml.parameterset.**Parameterset**(*name*, *description*, *uri=None*)
Bases: object

Parameter set

pycropml.parameterset.**parameterset**(*model*, *name*, *kwds*)

pycropml.pparse module

License, Header

class pycropml.pparse.**ModelParser**
Bases: *pycropml.pparse.Parser*

Read an XML file and transform it in our object model.

Algorithm(*elt*)

Description(*Title*, *Author*, *Institution*, *Reference*, *Abstract*)

Input(*elts*)

Inputs(*Input*)

ModelUnit(*elts*)
ModelUnit (Description,Inputs,Outputs,Algorithm,Parametersets, Testsets)

Output(*elts*)

Outputs(*elts*)
Ouputs (Output)

Parameterset(*elts*)

Parametersets(*Parameterset*)

Testset(*Test*)

Testsets(*Testset*)

dispatch(*elt*)

param(*pset*, *elt*)
Param

parse(*fn*)

class pycropml.pparse.**Parser**
Bases: object

Read an XML file and transform it in our object model.

dispatch(*elt*)

parse(*fn*)

pycropml.pparse.**model_parser**(*fn*)
Parse a set of models as xml files and return the models.

Returns ModelUnit object of the CropML Model.

pycropml.render_notebook module

pycropml.render_python module

pycropml.version module

Maintain version for this package. Do not edit this file, use 'version' section of config.

pycropml.version.**MAJOR = 0**
(int) Version major component.

pycropml.version.**MINOR = 0**
(int) Version minor component.

pycropml.version.**POST = 2**
(int) Version post or bugfix component.

Module contents

1.5 Usecases

1.6 Licence

PyCropML is released under a MIT License.

1.7 Glossary

Terminology

Model Simplified representation of the crop system within specific objectives.

CHAPTER 2

Credits

1. Development Lead

- Cyrille Ahmed Midingoyi, <cyrille.midingoyi@inra.fr>
- Christophe Pradal, <christophe.pradal@cirad.fr>

2. Contributors

None yet. Why not be the first?

CHAPTER 3

History

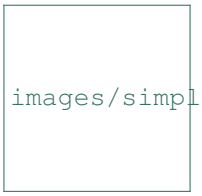
- First release on PyPI.

3.1 Indices and tables

- genindex
- modindex
- search

3.2 Supported by:


images/siriusquality.png


images/simplace.png

Python Module Index

p

Index