easy_install CIM

COMMON INFORMATION MODELS IN PYTHON

Richard. W. Lincoln

May 14, 2009

INSTITUTE FOR ENERGY AND ENVIRONMENT

Department of Electronic & Electrical Engineering

The University of Strathclyde

Outline

- CIMGym
  - Eclipse RCP application
- PyEC 61970
  - **Python implementation of the CIM**
- Pylon
  - Power Flow / **Optimal Power Flow**
- Pyreto
  - **Energy Market Simulation** Platform
  - Reinforcement Learning

Acknowledgements

- **Open source** software
- SUPERGEN: **Highly Distributed Power Systems** Consortium
  - Headed by Professor Graeme Burt
- **EPSRC** funded
- Engineering challenges associated with widespread adoption of distributed generation

- `http://www.supergen-hdps.org/`

CIMGym

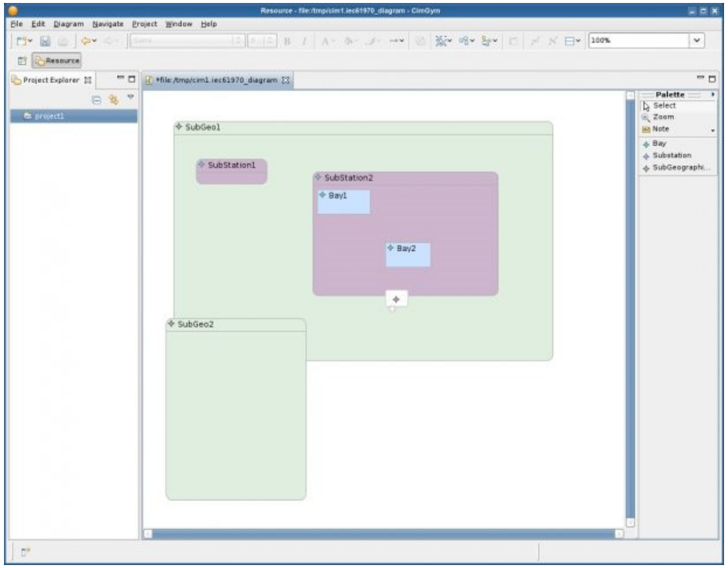
University of Strathclyde Engineering

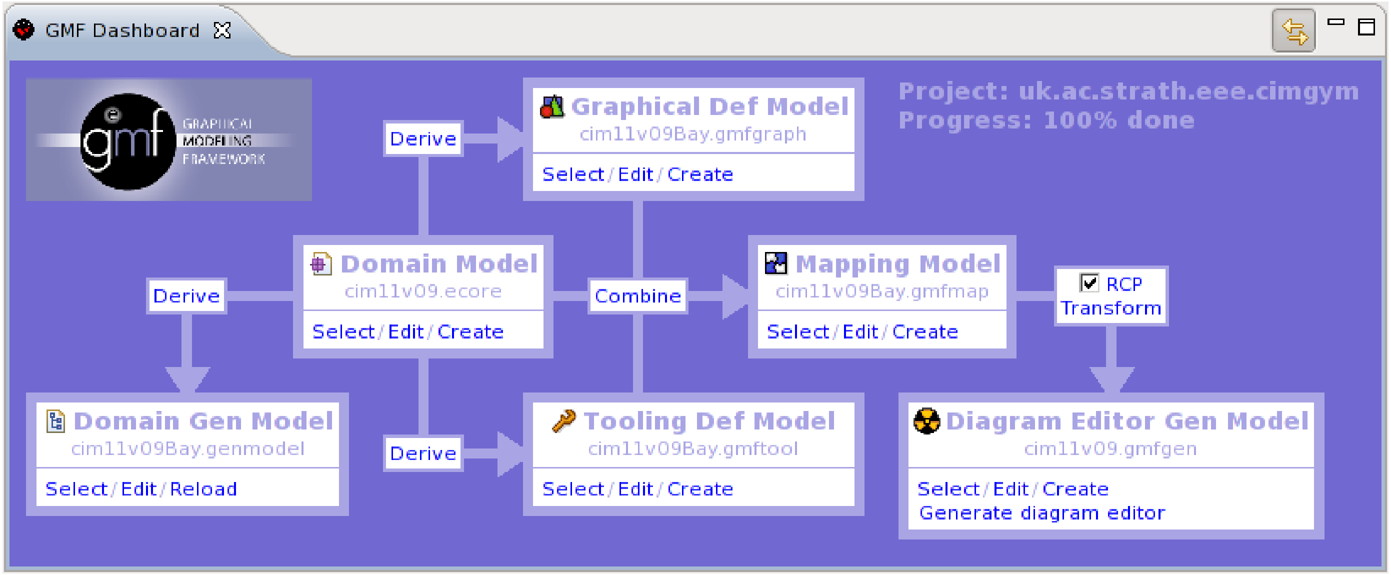
- Cross-platform **Desktop Application** built on the Eclipse Rich Client Platform
- **Generated** entirely from models
- Tree editor for CIM
  - Context menu
  - Properties View
- License: **GPL v2**
- `http://cimgym.sf.net`

Platform	Downloads
Linux (x86_64)	17
Linux (x86)	9
Mac OS X (x86)	6
Mac OS X (PPC)	2
Windows	83
Source	31
Total	**148**

CIMGym and GMF

- Eclipse Graphical Modeling Framework
- Map graphical and tooling definition models





PyEC 61970

- Implementation of the Common Information Model
  - **Python** programming language
  - Generated from Eclipse **Ecore model**
- Simple and approachable
  - For non-CS graduates

PyEC 61970

- Implementation of the Common Information Model
  - **Python** programming language
  - Generated from Eclipse **Ecore model**
- Simple and approachable
  - For non-CS graduates
- `$ easy_install CIM`
- `>>> from CIM.Wires import ACLineSegment`
- Use of the **Traits** project
  - Enhanced Python attributes

Traits

- Enhanced **Python attributes**
- Enthought Tool Suite
  - Enthought Inc. (NumPy, SciPy and IPython)
  - **Scientific & Engineering** applications
  - BSD-style licence
- Initialisation (default values)
- **Validation** (type checking)
- Notification (**event system**)
- **Visualisation** (tool-kit independent GUI)

PyEC 61970

- **Bi-directional associations** handled
  - Opposite trait **meta-data**
  - Event notification system
- Unique names for IdentifedObject
  - Trait initialisation
- CIM **RDF/XML parsing**
  - `rdfxml.py` by Sean B. Palmer (GPL v2)
  - Optimised **`rdflib`** with custom Store
- Serialise with Python's **`pickle`** library

PyEC GUI

- **Tool-kit independent** cross-platform GUI
  - **wxPython** and **PyQT** currently
  - **Tkinter** included with Python
  - **Pyjamas** (GWT for Python)
- Tree / table **editors**
- **Graph** editor
  - Python implementation of **Graphviz** API with Traits
  - Interactive (Context menu, Pan, Zoom etc.)
  - **HTML** labels
  - **SVG** shape files (WIP)

Desktop App

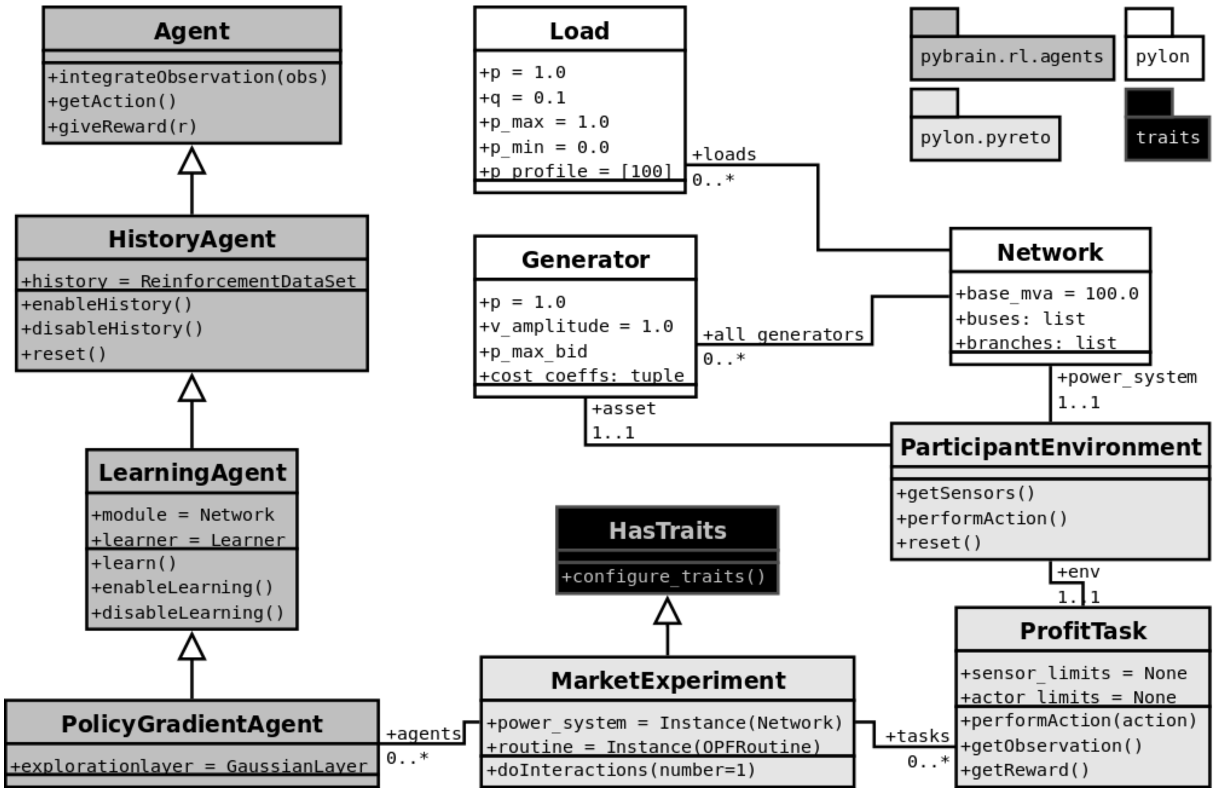
- **View Model** Traits application
  - Silo approach
- Envisage application framework
  - **Plug-in** structure
  - **Lazy** loading
  - Core plug-ins
    - Workbench plug-in (**View**, **editors** and **perspectives**)
    - Resource plug-in (File system resource access)
    - IPython plug-in (Scripting operations)
    - Logging plug-in

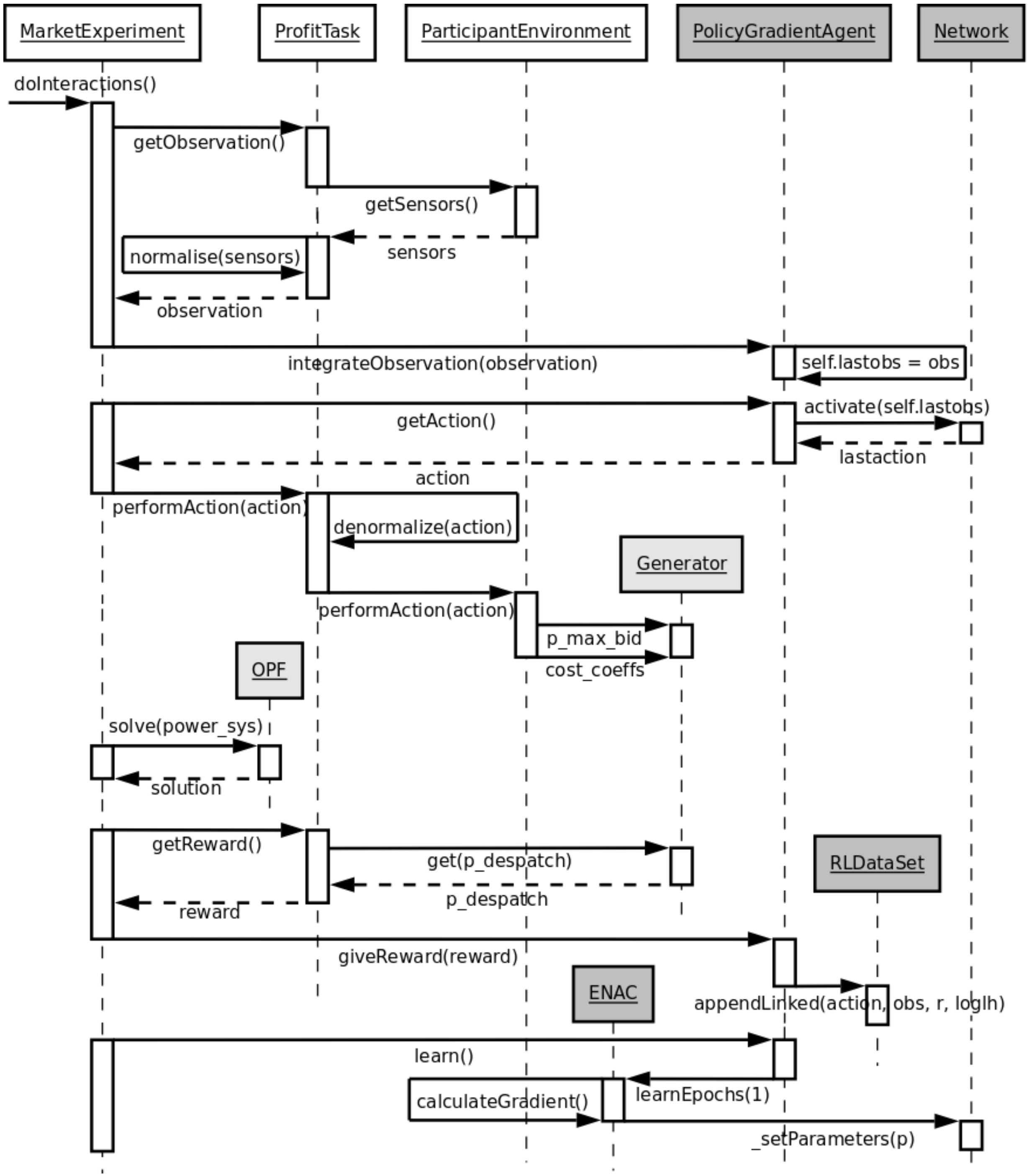
Pylon

- **Power system analysis** with Python
  - Bus – Branch model
  - Power Flow, **Optimal Power Flow** (DC/AC)
  - Routines from **MATPOWER** (PSERC)
- CVXOPT (Convex Optimisation)
  - **Sparse** and dense matrices
  - Interfaces to CHOLMOD, BLAS and MOSEK
  - Linear, quadratic and **non-linear solvers** in pure Python

Pyreto

- **Energy market** simulation extension to Pylon
  - OPF representing **centralised** market
  - **Reinforcement learning** agents




MarketExperiment
ProfitTask
ParticipantEnvironment
PolicyGradientAgent
Network
doInteractions()
getObservation()
getSensors()
sensors
normalise(sensors)
observation
integrateObservation(observation)
self.lastobs = obs
getAction()
activate(self.lastobs)
lastaction
action
performAction(action)
denormalize(action)
Generator
performAction(action)
p_max_bid
cost_coeffs
OPF
solve(power_sys)
solution
getReward()
get(p_despatch)
p_despatch
reward
RLDataSet
giveReward(reward)
ENAC
appendLinked(action, obs, r, loglh)
learn()
learnEpochs(1)
calculateGradient()
_setParameters(p)

EEM '09

- Conference Paper
- 6th International Conference on the **European Energy Market**
- Katholieke Universiteit **Leuven**
- May 27-29, 2009

Open Source, Agent-based Energy Market Simulation with Python

Richard W. Lincoln, *Student Member, IEEE* Stuart Galloway, Graeme Burt, *Member, IEEE*

***Abstract*—Increasingly, the electric energy transmitted and distributed by national power systems is traded competitively in free markets. Long-term decisions must be made by authorities as to the structure of energy markets and the regulations that govern interactions between participants. It is not practical to experiment with real energy markets and in order to establish the potential effects of making these decisions there are few options but to simulate the markets computationally. This paper proposes that the complexity of power systems and the associated energy markets necessitates an open approach in their modelling and simulation. It presents an open source software package for simulating electric energy markets using the Python programming language. Power systems and their associated constraints are modelled using traditional steady-state analysis techniques. While market participants are represented by reactive agents that learn through reinforcement. The software and all of its dependencies are open and freely available to the scientific community.**

***Index Terms*—Steady-state simulation, Energy markets, Agent-based simulation, Open source software, Reinforcement learning, AC Optimal Power Flow**

I. Introduction

ELECTRIC energy use pervades almost every aspect of life in an industrialised society and as such a myriad of factors influence the balance of demand and supply. Both technically and operationally, no practical inroads have yet to be made in forecasting market prices. Nevertheless, electricity is traded competitively on a national scale on a day to day basis. Large sums of money change hands in the world's energy markets and the benefits of competitive trade to society can be great. Small improvements in market design or strategic participation can benefit social welfare and conversely so.

Relative to most other commodities, trade of electric energy is still in its infancy. When competition was first introduced, the unique nature of electricity as a commodity raised many doubts as the whether such a vital resource for modern societies could be reliably traded in the free market. Important decisions were made when the trading arrangements for early energy markets were first decided upon and had to be supported through research where possible[1]. Liberalisation and unbundling of electricity supply industries costs many millions of pounds to implement. Countries, having made this investment, continue to restructure and adjust their energy markets in the hope of further reducing costs to the consumer and promoting innovation and efficiency through competition.

Manuscript received 20090309
R. W. Lincoln, S. Galloway and G. Burt are with the Department of Electronic and Electrical Engineering, The University of Strathclyde, Glasgow, Scotland, G1 1XW UK e-mail: rlincoln@eee.strath.ac.uk.
Digital Object Identifier goes here

Electric energy is key to the operation of industrialised societies. It is not practical to experiment with real energy markets and in order to establish the potential effects of particular changes to rules and regulations there are few options but to attempt to simulate them computationally.

Game theoretic models are commonly associated with economics and attempt to capture behaviour in strategic situations mathematically. They have been applied to electric energy problems of many forms, including but not limited to analysis of market structure, market liquidity, pricing methodologies, regulatory structure, plant positioning and network congestion. More recently, agent-based simulation has received a certain degree of attention from researchers and has been applied in some of these fields also.

While popular and seemingly promising, agent-based simulation is still centred around abstracted models. The assumptions made is this abstraction must be subjected to the same verification and validation as with equation-based models. Verification of assumptions and model validation are often overlooked in agent-based simulations of energy markets, yet they are possibly the most important steps in the model building process. Techniques used to develop, debug and maintain large computer programs can often be used to verify that a model does what it is intended to do.

Validation of an energy market model is more difficult. It can be accomplished using the intuition of experts or through comparison of simulation results with either historical market data or theoretical results from more abstract representations of the model. Finding verifyable trends in existing markets is a very large challenge. To then prove that a computational model replicates these characteristics with suitable fidelity is yet more challenging still. Only when a model is suitably verified and validated can any conclusions be drawn from results obtained through implementation and simulation of suitable scenarios.

To accelerate knowledge and understanding in this important and challenging field, the scientific community requires a more *open* approach. Possibly for commercial reasons, all but one[2] of the recently developed energy market simulation programs remains closed source. While the features of these simulators and the methods used in implementing them are documented to some extent in their associated publications, no access is provided to the source code for the software. As with all research related to closed source software projects, this prevents third parties from scrutinising the intricacies of the implementation and building directly upon previous effort.

This paper presents an agent-based energy market simulation package developed as an extension to an existing power systems analysis package. It is written in a general purpose, http://pylon.eee.strath.ac.uk

http://github.com/rwl

http://cimgym.sf.net

richard.lincoln@eee.strath.ac.uk