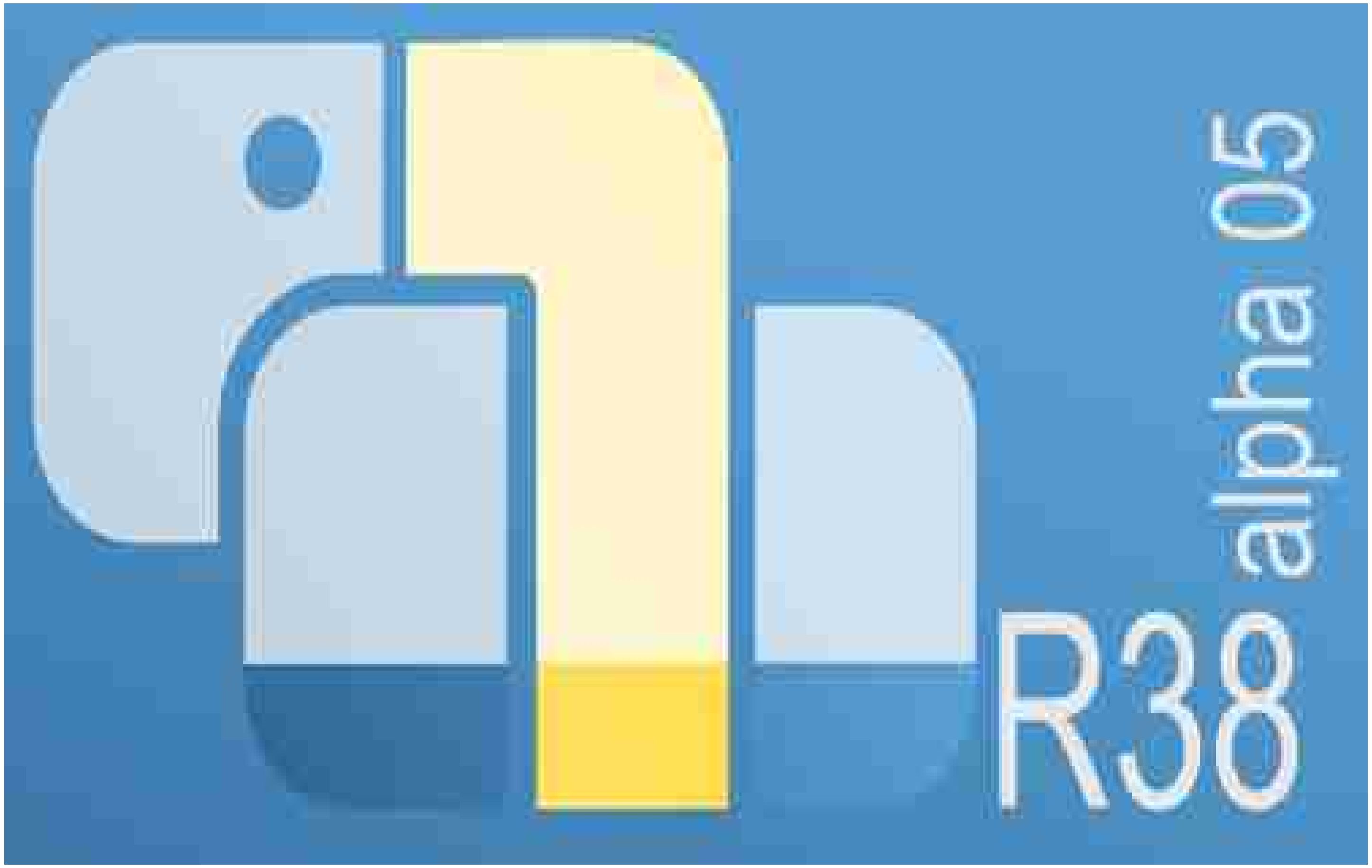
PythonCAD

… A Python Story



Once upon a time ...

A little guy that was fond of programming

```
    **** COMMODORE 64 BASIC V2 ****
 64K RAM SYSTEM  38911 BASIC BYTES FREE
READY.
```

Started to write his first lines of code trying to develop simple games and database

Little guy grew up

Learning QBasic, Fortran, vb6, c# … were interesting BUT…

all this program were not free enough and they needed a compiler .... as I'm lazy ...

Where can I find a free, simple and powerful programming language?

....

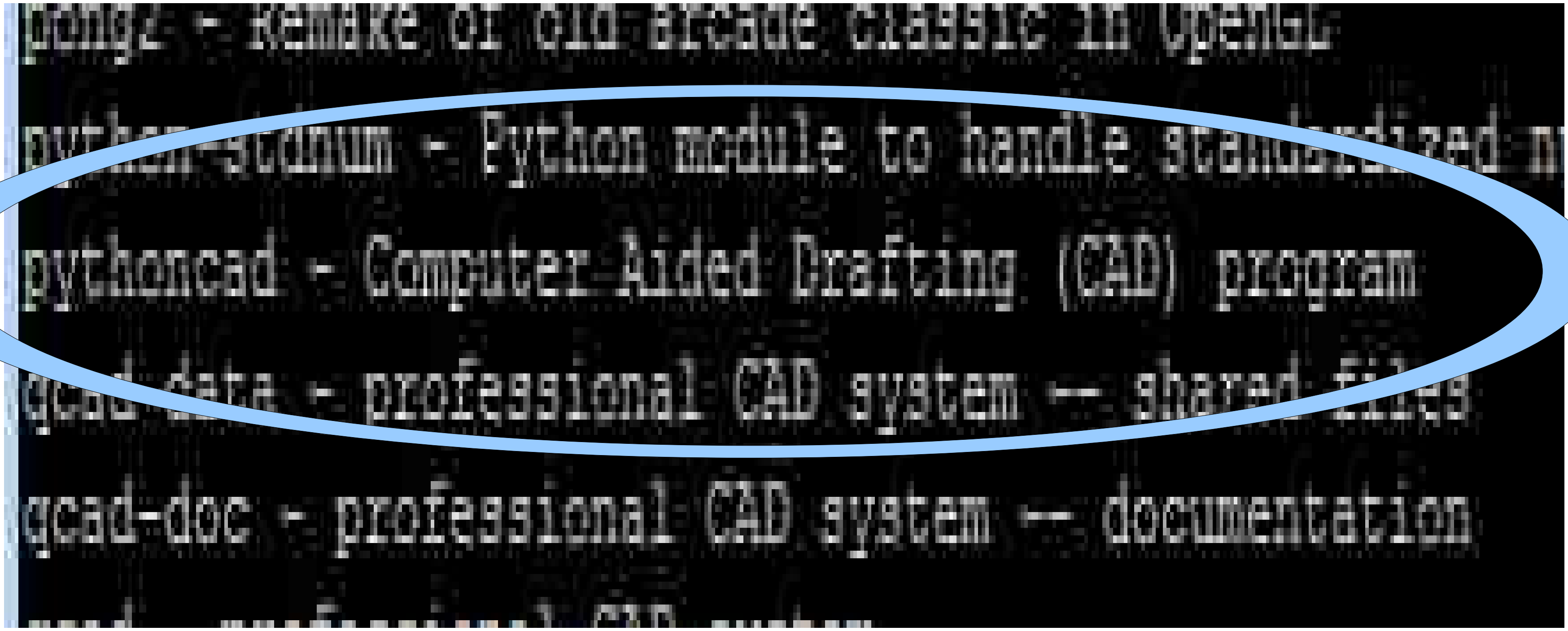
The New World

It's also cross platform ..that's cool !!

! I have an old pc … so I can run Linux on it … Debian is great .. full of packages and completely OpenSource!

Why PythonCAD ??

$ apt-cache search cad



```
pong2 - Remake of old arcade classic in OpenGL
python-stdnum - Python module to handle standardized n
pythoncad - Computer Aided Drafting (CAD) program
qcad-data - professional CAD system -- shared files
qcad-doc - professional CAD system -- documentation
```

I love Python ...I work for a cad company... I like cad …
Why not doing some little improvements at the project ???!??!?!

PythonCAD Story

Art Haas Ideator and first developer

- Work began on PythonCAD in July, 2002, and the first public release was on December 21, 2002.
- The thirty-sixth release of PythonCAD was made available on May 12, 2007

How it was ..

- Gtk Client interface
- No support for other format like (dxf,dwg)
- Xml file format
- No public repository to make any contribution
- Over 2 years of inactivity

In Few word ..

PythonCAD was ..

But Software never die !!

- Like a Phoenix can be taken to life again
- I get in tuch with Art
- Got the code ..
- I opened a new sourceforge Project to get more visibility and attract new developers

The work starts !!

- Start developing new functionality
  - Dinamic view
  - Snap
  - And a new version R37 ....

Keep in touch with PythonCAD Debian maintainer to update the version (that is the actual one on Debian\Ubuntu)

Something Append...

- I found a new developer that want to implement the dxf import/export ..
- As soon we load the first big drawing ..form dxf
- Python show the first performance issue ..
- Very slow on pan and zoom operation... (very bad issue for a 2d Cad package !)

PythonCAD - Python truble

- Gtk is very good but to render the entity at each pan operation python looped up to show the visible entity...

- Memory issue

So only c++ is good enough for a cad package ??

Testing new interface

- Cairo
- WxPython
- Tkinter
- PyQt

Looking deeper at PythonCAD

- Structure is not good …
- Interface is mixed with kernel ..
- File format is very bad for extending new entity

The new PythonCAD Generation

- New way to manage the file using sqlite
- New way to compute geometrical operation using sympy
- New interface using qt
- New code structure

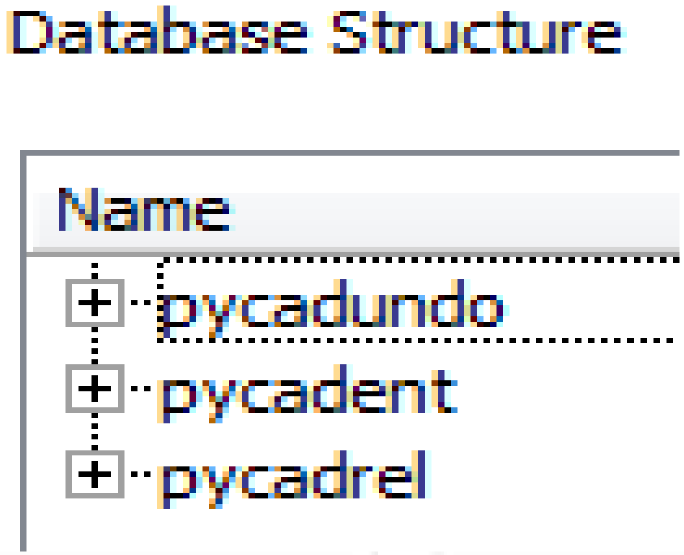
Sqlite

- Store information in file or in memory
- Embedded in python

If we use db we can even revisioning the drawing we have unlimited undo history we can manage relations...
That's really amazing ...

Are you sure that it works ??

Db Stucture



Database Structure

Name
- pycadundo
- pycadent
- pycadrel

Only 3 Tables
- Undo history
- Entity
- Entity relation
- cpickle geoEntity
- cpickle for style

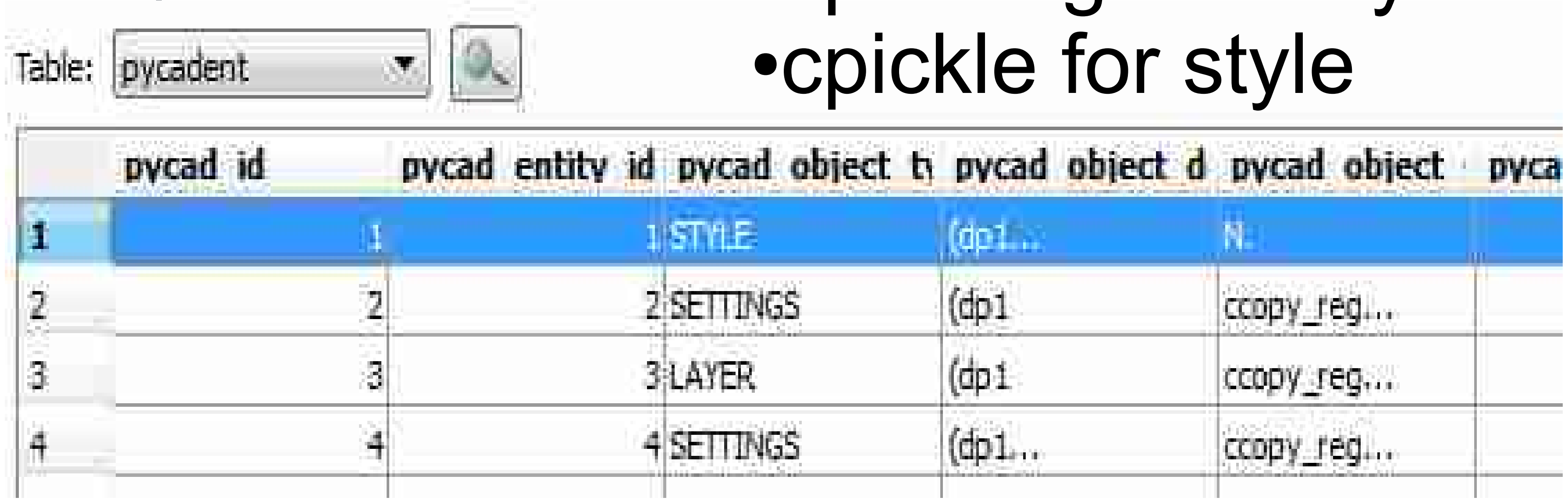


Table: pycadent

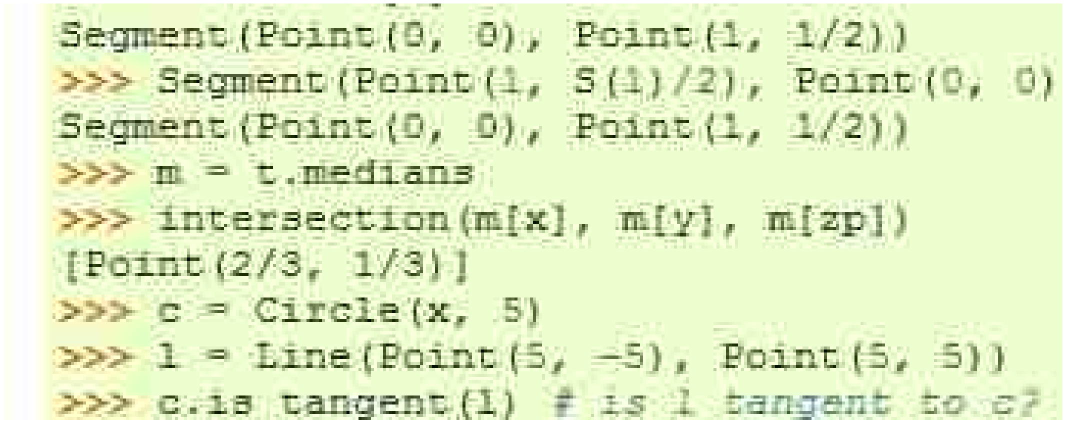
	pycad id	pycad entity id	pycad object ty	pycad object d	pycad object	pyca
1	1	1	STYLE	(dp1...	N.	
2	2	2	SETTINGS	(dp1	ccopy_reg...	
3	3	3	LAYER	(dp1	ccopy_reg...	
4	4	4	SETTINGS	(dp1...	ccopy_reg...	

SymPy

SymPy is an open source Python library for symbolic mathematics. It aims to become a full-featured computer algebra system (CAS) while keeping the code as simple as possible in order to be comprehensible and easily extensible. SymPy is written entirely in Python and does not require any external libraries.

- I just need to transform pyCAD entity into sympy entity and all the geometrical operation will be ok

- With sympy it's easy to perform unit mesure transformation

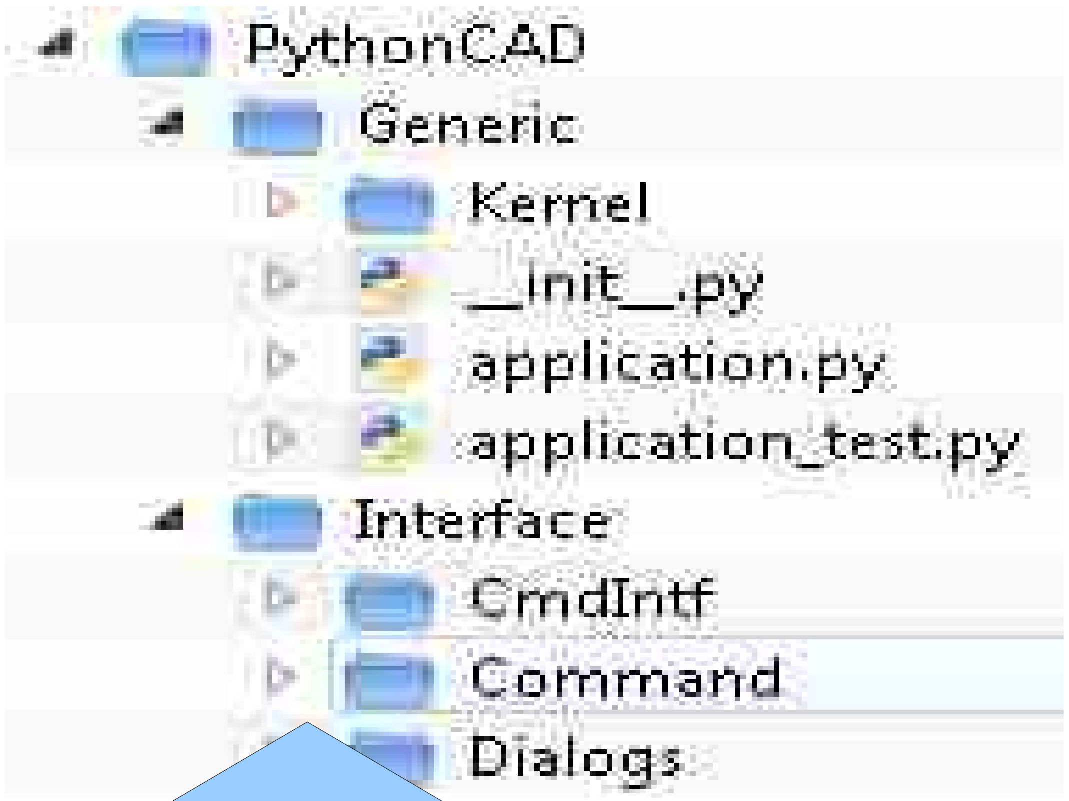


```
Segment(Point(0, 0), Point(1, 1/2))
>>> Segment(Point(1, S(1)/2), Point(0, 0))
Segment(Point(0, 0), Point(1, 1/2))
>>> m = t.medians
>>> intersection(m[x], m[y], m[zp])
[Point(2/3, 1/3)]
>>> c = Circle(x, 5)
>>> l = Line(Point(5, -5), Point(5, 5))
>>> c.is_tangent(l) # is l tangent to c?
```

PyQT

- It's Cross platform
- We have a good ide eric
- The QtGui.QGraphicsView and QtGui.QGraphicsScene store and render the entity
- We have a compiler ... PyInstaller for windows user
  - python Makespec.py -n PythonCAD --icon iconPath -p ModulePath ApplicationFilePath
  - python Build.py pathToSpecFile

New code structure



```
PythonCAD
    Generic
        Kernel
        __init__.py
        application.py
        application_test.py
    Interface
        CmdIntf
        Command
        Dialogs
```

Kernel

•Manage all geometrical entity

•Saving file

•Make geometrical operations

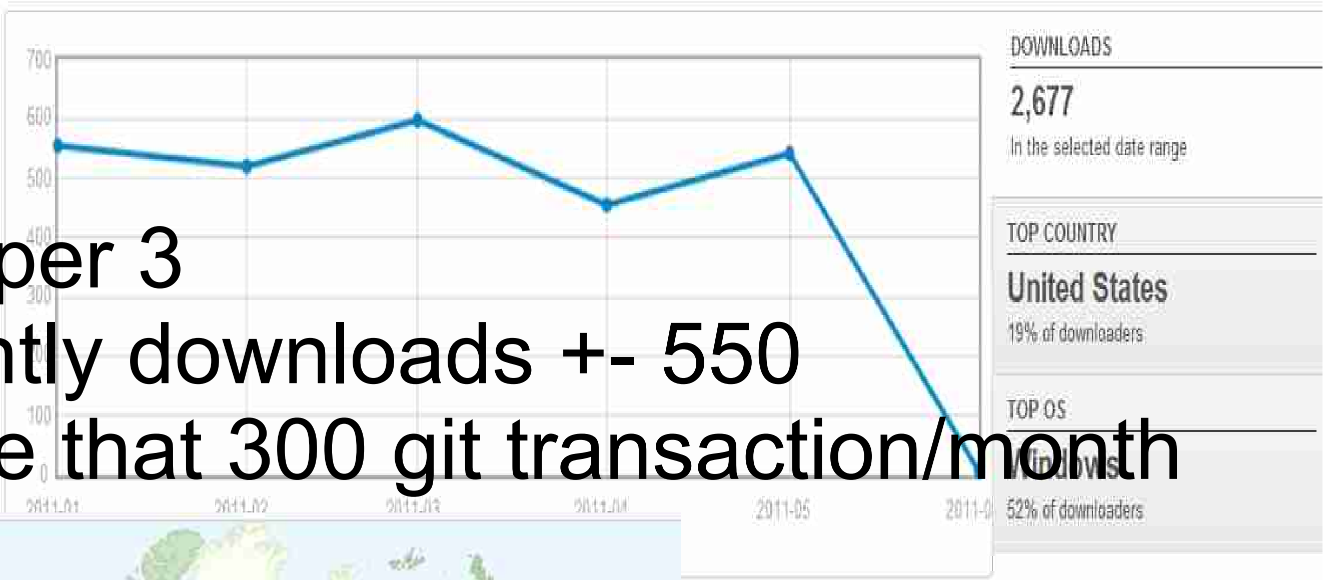
•Indipended from the interface

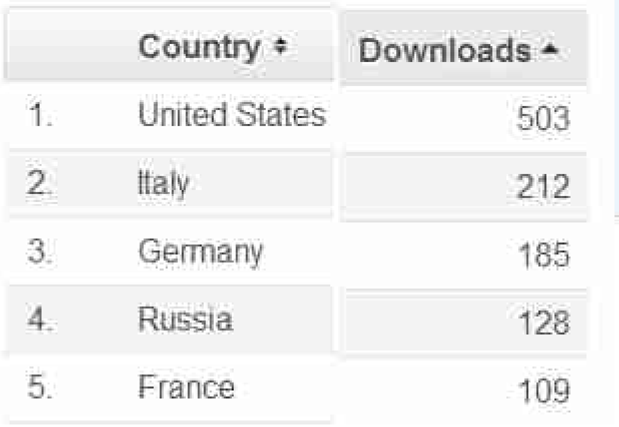
Interface

•Render the entity
•Manage the command
•Get informations from the user

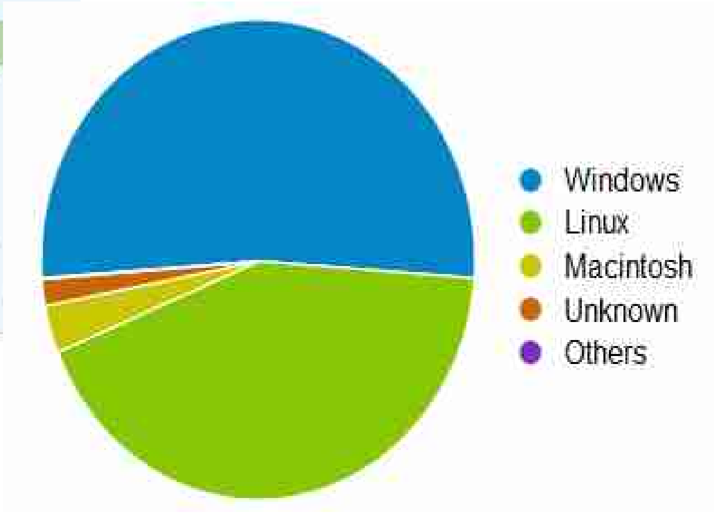
Project Statistic

- Active developer 3
- Windows montly downloads +- 550
- We have more that 300 git transaction/month





	Country	Downloads
1.	United States	503
2.	Italy	212
3.	Germany	185
4.	Russia	128
5.	France	109



Road Map For the first beta

- Fix Bugs
- Dimension style
- Hatch
- Preview
- Dwg read\write using libreDwg

Feature

- Unlimited Undo
- Revision history
- Import export in different format dwg\dxf
- Cross platform
- Easy to extend (it's all python)

......

- Manage groups
- Extend with mecanical feature
- Extend with architectural feature
- Any idea is welcome ....
- We need contribution to go on with this project

But the story goes on ...

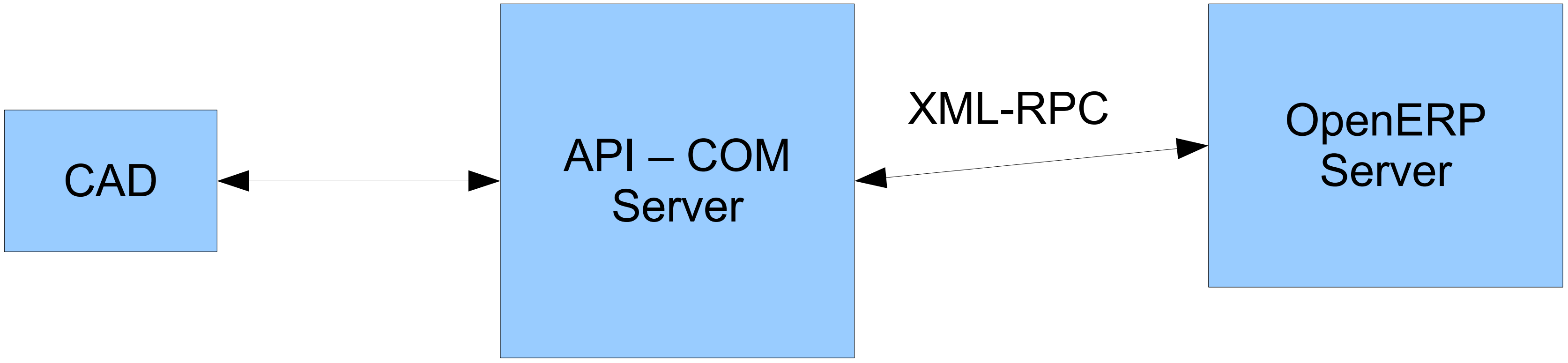
- In October 2010 I left my company
- Open my own Company OmniaSolutions.eu
- Became OpenErp partner at the end of 2010
- Start developing a new OpenErp Module
- OpenErp PLM

OpenERP PLM

- Full feature PLM integrated with OpenERP
- Full integrated with Thinkdesign (CAD From think3.com)
- It allows you to directely write the product data in OpenERP

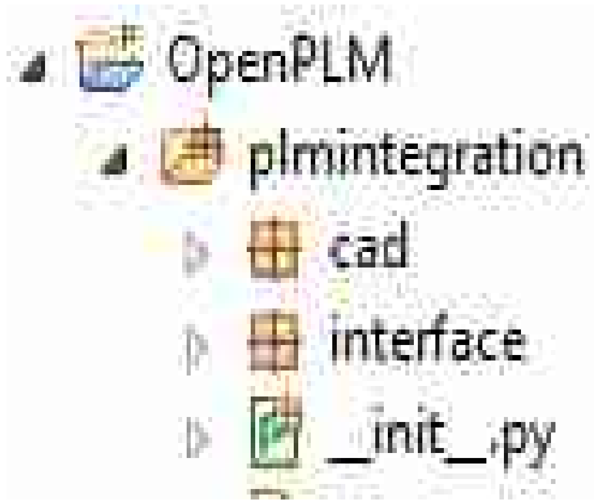
Tecnical Feature

- Using Koo Framework (PyQT) to integrate cad application with OpenERP
- Using XML-RPC to comunicate with OpenERP
- Create a ComApi server to integrate the python code with the cad application



Timing ..

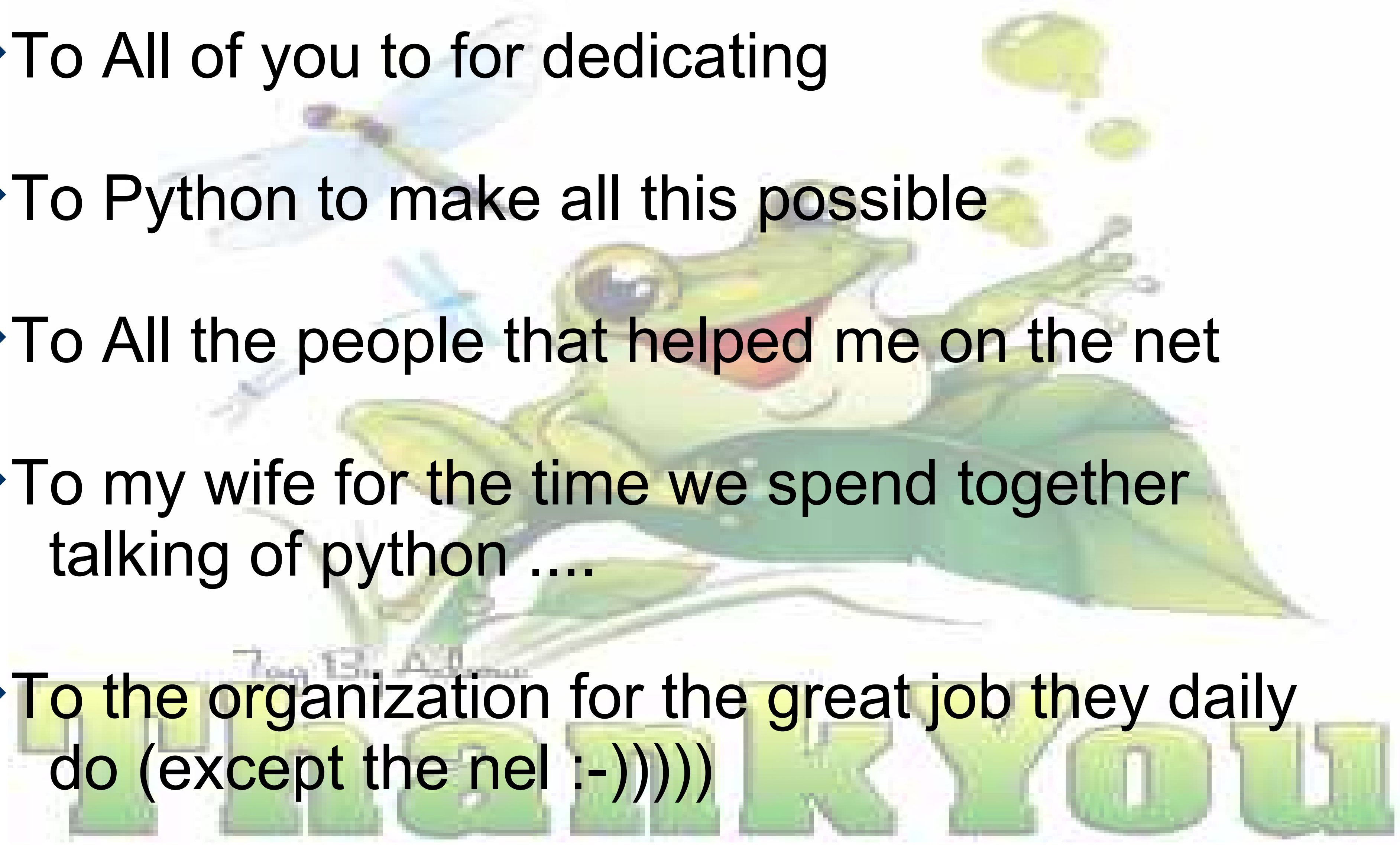
- It Takes less then 4 month in 1 people working time to build the first full feature working Alfa
- 1 more month to go on Beta
- Now we are in production with this solution


OpenPLM
plmintegration
cad
interface
init.py

So ..

- Python is the real actor of this solution
- Easy to learn
- Solid
- Have many idle / editor to feet the programmer needs
- There is at least one module for your needs
- Cross Platform

Thank ..



- To All of you to for dedicating
- To Python to make all this possible
- To All the people that helped me on the net
- To my wife for the time we spend together talking of python ....
- To the organization for the great job they daily do (except the nel :-)))))

.. Any questions ..

http://sourceforge.net/projects/pythoncad/ matteo.boscolo@omniasolutions.eu