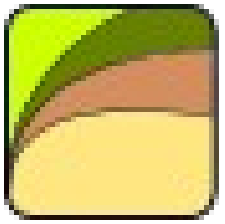


atmantree.com

En caso de dudas escriba a:
info@atmantree.com

Persistencia de Datos con ZODB

Carlos Gustavo Ruiz
@atmantree
Noviembre 2013

Agenda

- Presentación
- Persistencia de Datos
- Gestores Bases de Datos
- Python
- Bases de Datos en Python
- ZODB
- Escalar ZODB
- Palabras Finales

Presentación

yo = Persona()

yo.nombre = “Carlos Gustavo Ruiz”

yo.alias = “arahat”

yo.profesion = “Ingeniero en Sistemas”

yo.blog = “http://atmantree.com”

yo.twitter = “@atmantree”

yo.github = “http://github.com/atmantree/”

yo.roles = [“Consultor”, “Programador”, “Voluntario”]

dbroot['11am'] = yo

transaction.commit()

Persistencia de Datos

- Desde la antigüedad se sabe que la memoria es frágil y es difícil recordar cosas cada vez más complejas
- Aunque hay sus excepciones:

"Entrenando catorce horas al día, Jaime García (un colombiano que vive en Brunete) ha recitado 150.000 dígitos de π en la Facultad de Matemáticas de la Universidad Complutense de Madrid, batiendo el anterior récord del mundo (100.000 dígitos) de Akira Haraguchi estuvo rodeado de cierta polémica y ni siquiera fue considerado válido"

Microsiervos (http://is.gd/VH5YLn)

Para ver otros records visite:
http://www.recordholders.org/en/list/memory.html

Persistencia de Datos

- O excepciones de excepciones:

Persistencia de Datos

Sin embargo, desde el punto de vista de la informática y la computación es importante poder contar con los valores almacenados en un grupo de variables luego de finalizada la ejecución de un programa.

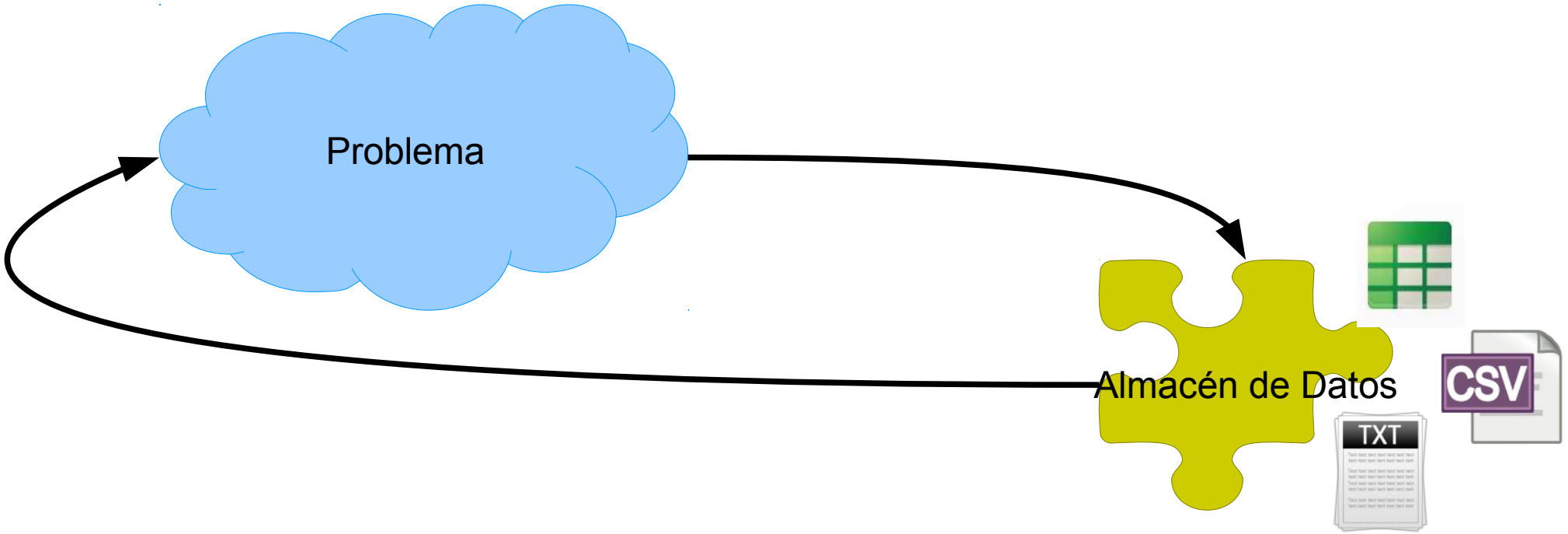
Persistencia de Datos

A la capacidad de mantener un dato luego de finalizada la ejecución de un programa (o de un determinado módulo del mismo) se le conoce como:

Persistencia de Datos

Gestores de Bases de Datos

Cuando utilizamos almacenes de datos (gestionar la persistencia a bajo nivel) siempre estamos muy cerca de la implementación del mecanismo de persistencia. No hay capacidad de abstraerse al dominio del problema.



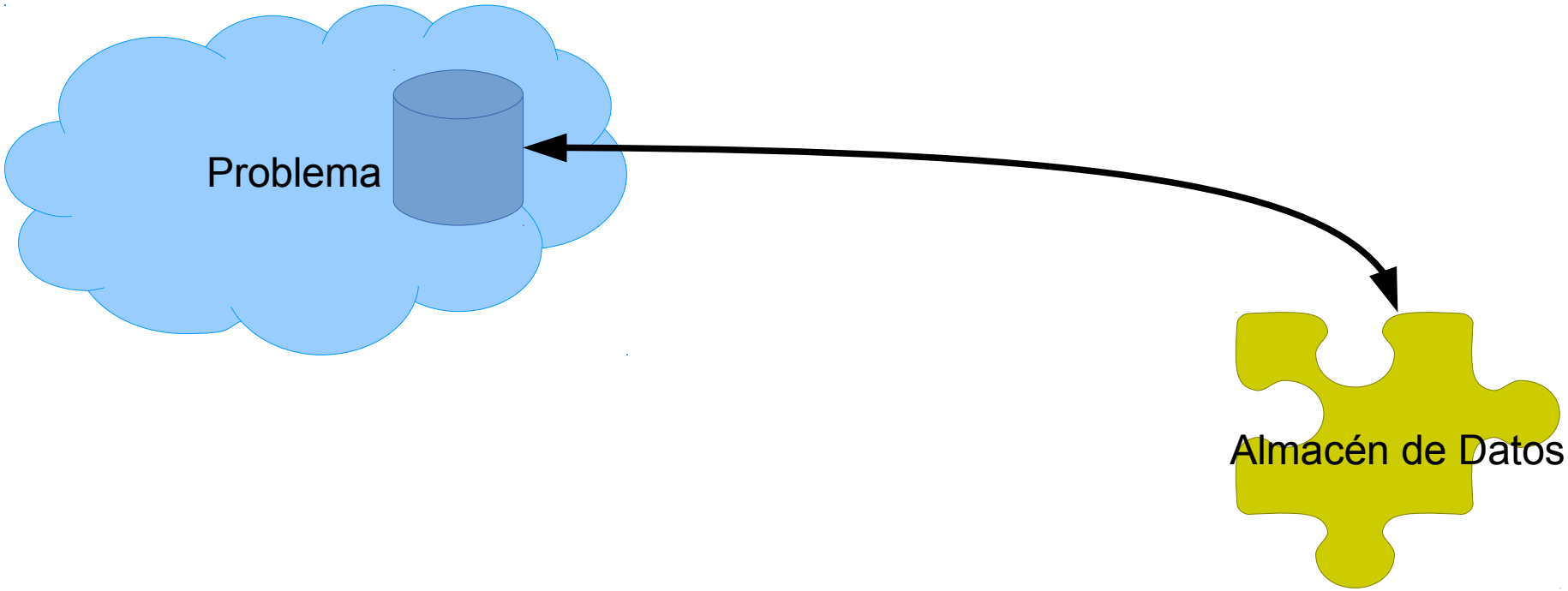
Gestores de Bases de Datos

Los gestores de bases de datos son una capa de abstracción que permite que nos concentremos en el problema y bajo ciertas condiciones confiemos a ellos el guardado y recuperación de la información.

En otras palabras los gestores de bases de datos nos permiten delegar la implementación de los almacenes a un programa.

Gestores de Bases de Datos

Esquemáticamente podemos decir en que nuestros datos hablan el mismo idioma que nuestro problema. Por lo tanto siempre conversamos al mismo nivel y en el mismo lenguaje.



Python

Python es un lenguaje de programación:

- Interpretado
- Dinámico
- Orientado a Objetos
- Fuertemente y Dinámicamente Tipado
- Con Baterías Incluídas
- Popular

Python

“Python es amor a la primera línea”
@iferminm

Bases de Datos en Python

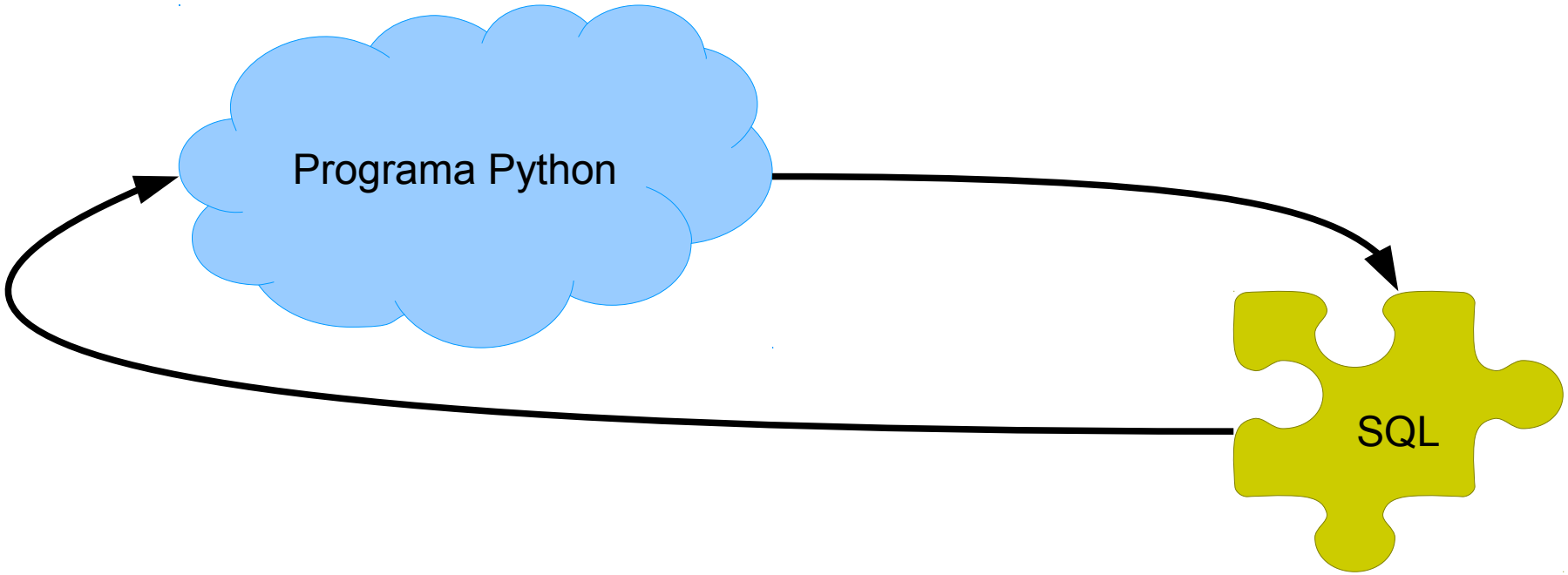
En general, hasta hace muy poco tiempo la mayor cantidad de bases de datos disponibles era del tipo relacionales. Por ende SQL es un mecanismo estándar para usar una base de datos.

Es por ello que Python en su PEP 249 estandariza el uso de base de datos mediante un mecanismo de conexiones y cursores.

http://www.python.org/dev/peps/pep-0249/

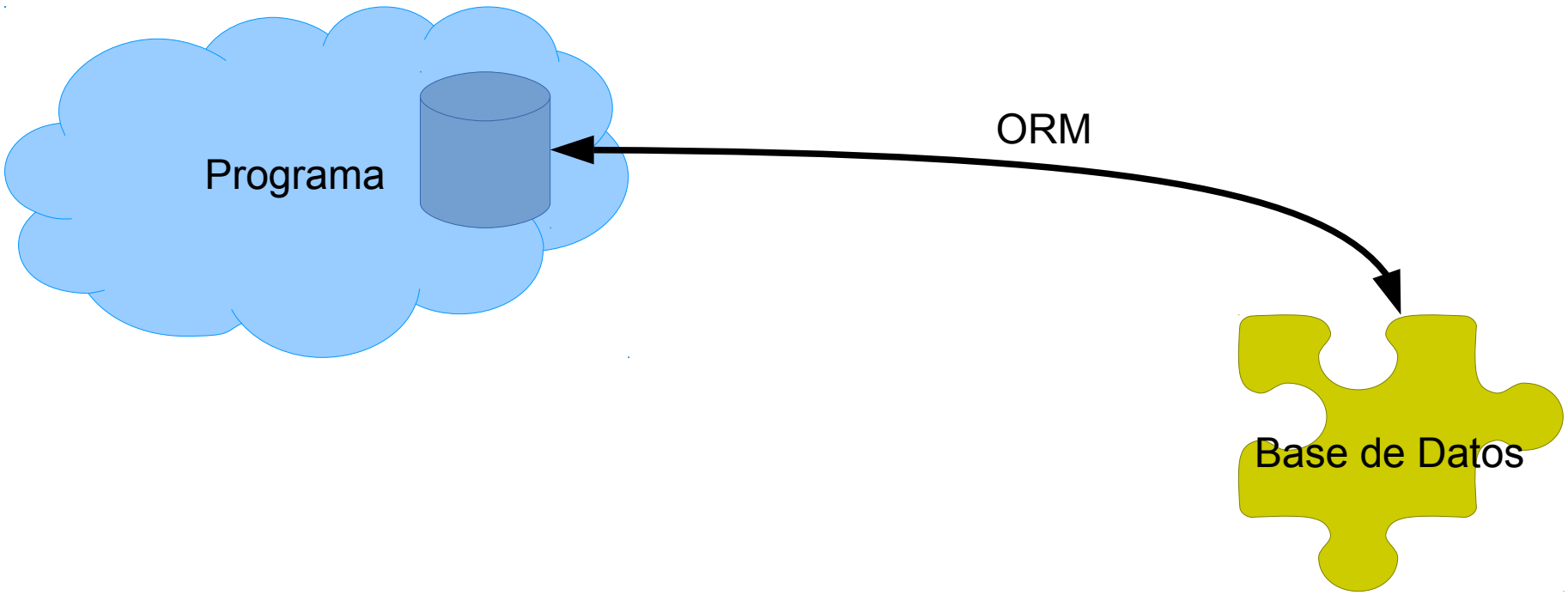
Bases de Datos en Python

Y en ese momento te conviertes en políglota, pues debes escribir en SQL dentro de tu aplicación Python.



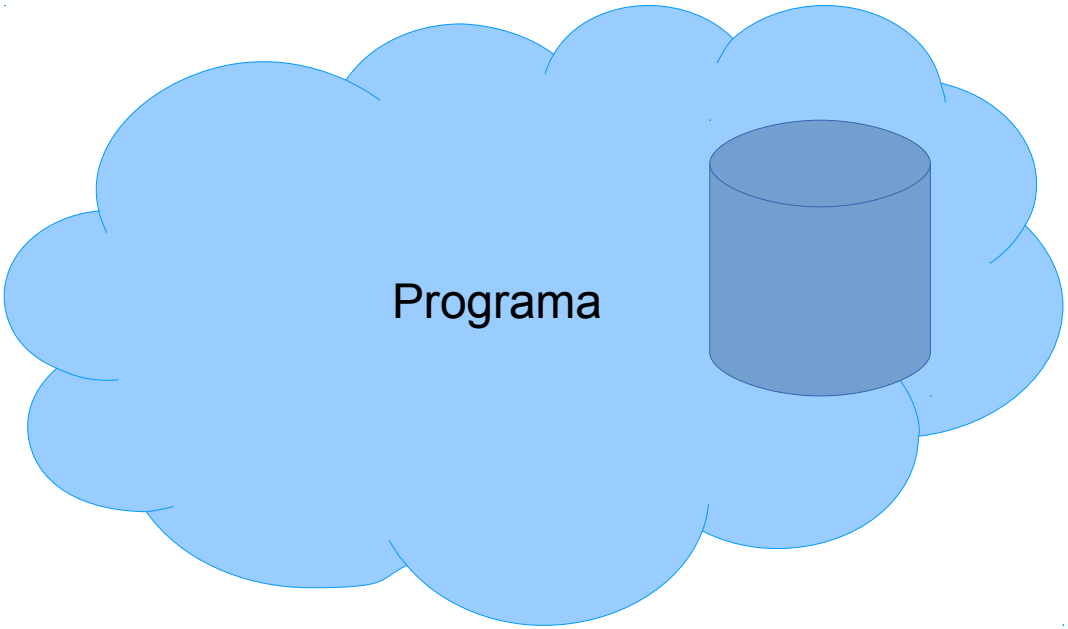
Bases de Datos en Python

Sin embargo hay módulos que permiten conversar con la base de datos hablandoles en Python. Ellos son los mapeadores objeto relacionales



Bases de Datos en Python

Sin embargo, ¿Qué tal si no tuviesemos que salir en absoluto de Python para persistir los datos de nuestra aplicación?



ZODB

ZODB → Z Object Database

ZODB le ofrece:

- Persistencia transparente de objetos Python
- Transacciones completamente compatibles con ACID
- Capacidad de Historiales y deshacer
- Almacenamiento eficiente de datos binarios (BLOBs)
- Almacenes empotrables
- Arquitectura escalable

ZODB

Instalar ZODB es simple:

```
$ pip install zodb
```

ZODB

Conversar con ZODB es tan fácil como usar el intérprete interactivo de Python:

```
>>> # llamado a los paquetes
>>> from ZODB import DB, FileStorage
>>> import transaction
>>>
>>> # se define la fuente física donde se encuentran los datos
>>> almacen = FileStorage.FileStorage("Datos.fs")
>>> bd = BD(almacen)
>>> con = bd.open()
>>> r = con.root()
>>>
>>> # ahora a guardar información
>>> r['saludo'] = 'Hola ZODB'
>>> r['numero'] = 3 + 7j
>>> r['lista'] = [1, 3, 5.987, True, 'texto']
```

ZODB

Conversar con ZODB es tan fácil como usar el intérprete interactivo de Python:

```
>>> # para ver las llaves que tenemos almacenadas
>>> r.keys()
['lista', 'saludo', 'numero']
>>>
>>> # almacenamos los cambios en el archivo de almacén
>>> transaction.commit()
>>>
>>> # cerramos la conexión.
>>> transaction.get()
>>> transaction.abort()
>>> con.close()
>>> bd.close()
>>> almacen.close()
```

ZODB

ZODB persiste como un conjunto de objetos relacionados a una raíz, al estilo de una base de datos de llave-valor, sin embargo los objetos pueden crear jerarquías

Debido a que ZODB es capaz de persistir cualquier objeto Python es interesante poder persistir objetos creados por un programa.Para ello las clases creadas en Python deberán ser subclases de Persistent.

Veamos un ejemplo

ZODB

Si deseamos crear una clase Tareas para una lista de recordatorios entonces deberíamos hacer algo como esto:

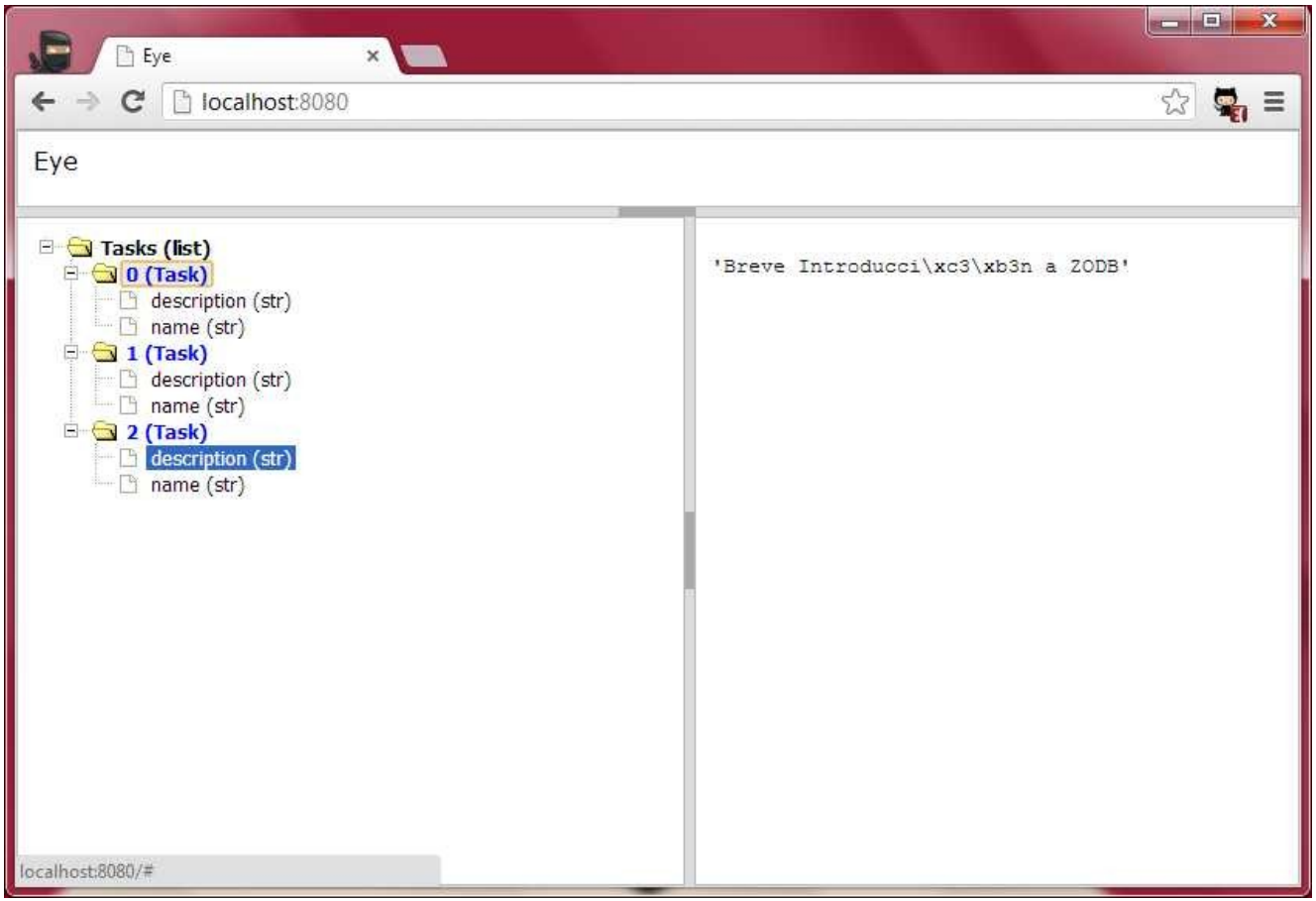
```
from ZODB import (DB, FileStorage)
from persistent import Persistent
import transaction
import argparse

class Task(Persistent):
    def __init__(self):
        self.name = ""
        self.description = ""
```

Y lo demás es similar a lo visto anteriormente.

ZODB

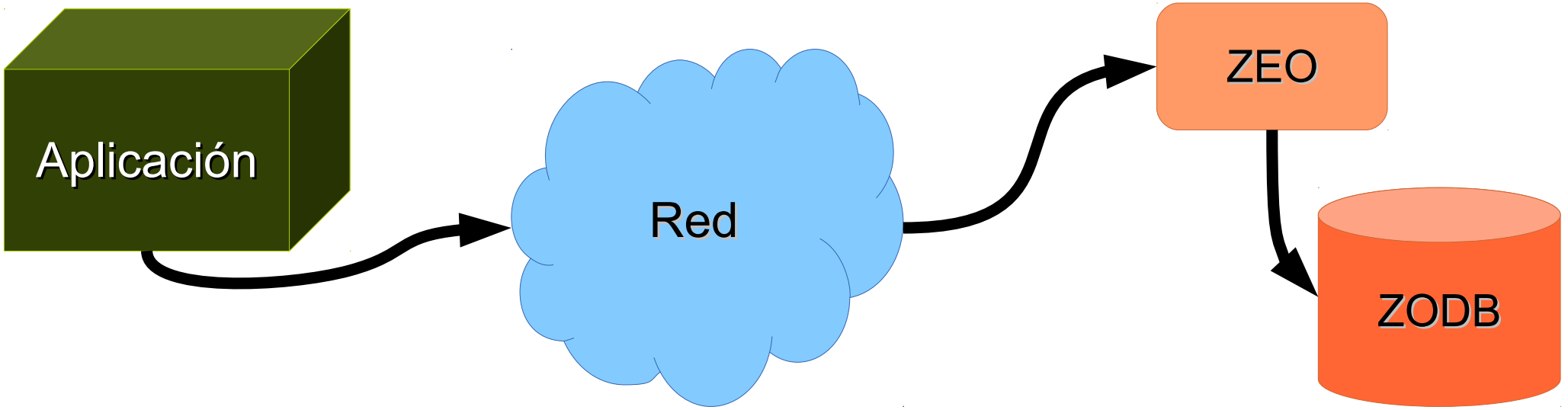
Existen herramientas gráficas para visualizar que contienen una base de datos ZODB.



Escalar ZODB

ZODB puede ser usado solamente por un solo proceso de Python. Para extender el funcionamiento de ZODB permitiendo el acceso de múltiples procesos y hacerlo desde la red se creo ZEO.

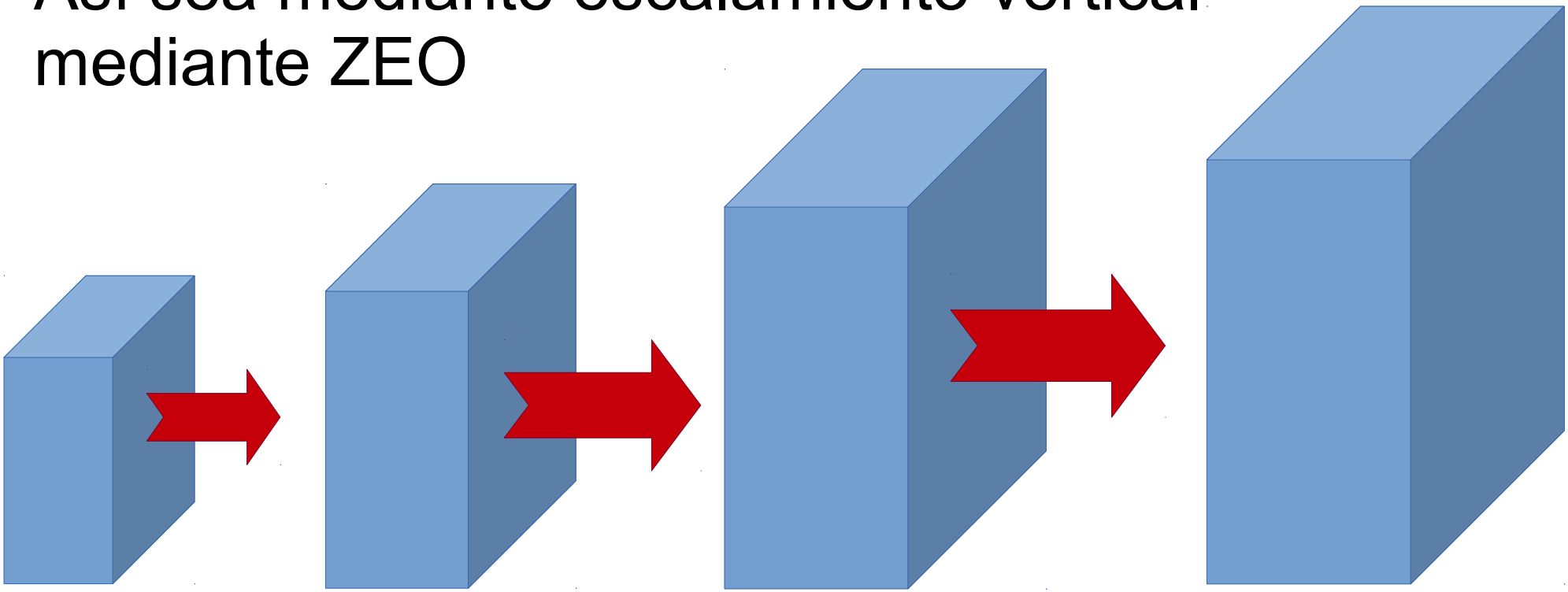
ZEO → Zope Enterprise Objects



Escalar ZODB

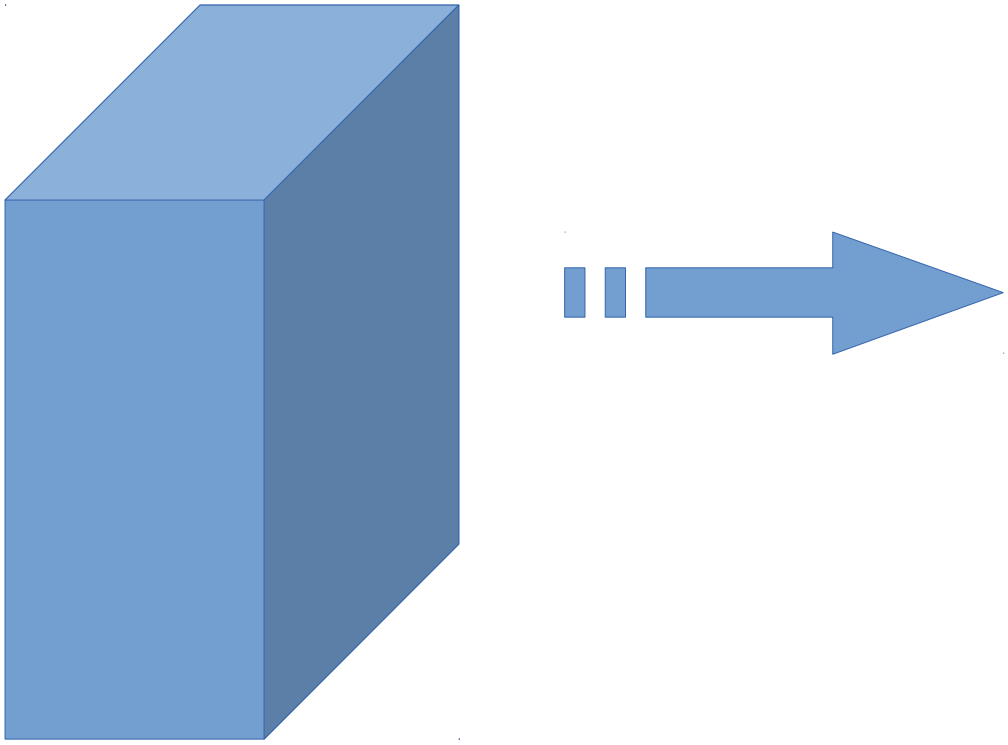
ZODB no limita el crecimiento de su aplicación, pues siempre podrá acompañarle con una solución a su necesidad específica.

Así sea mediante escalamiento vertical mediante ZEO



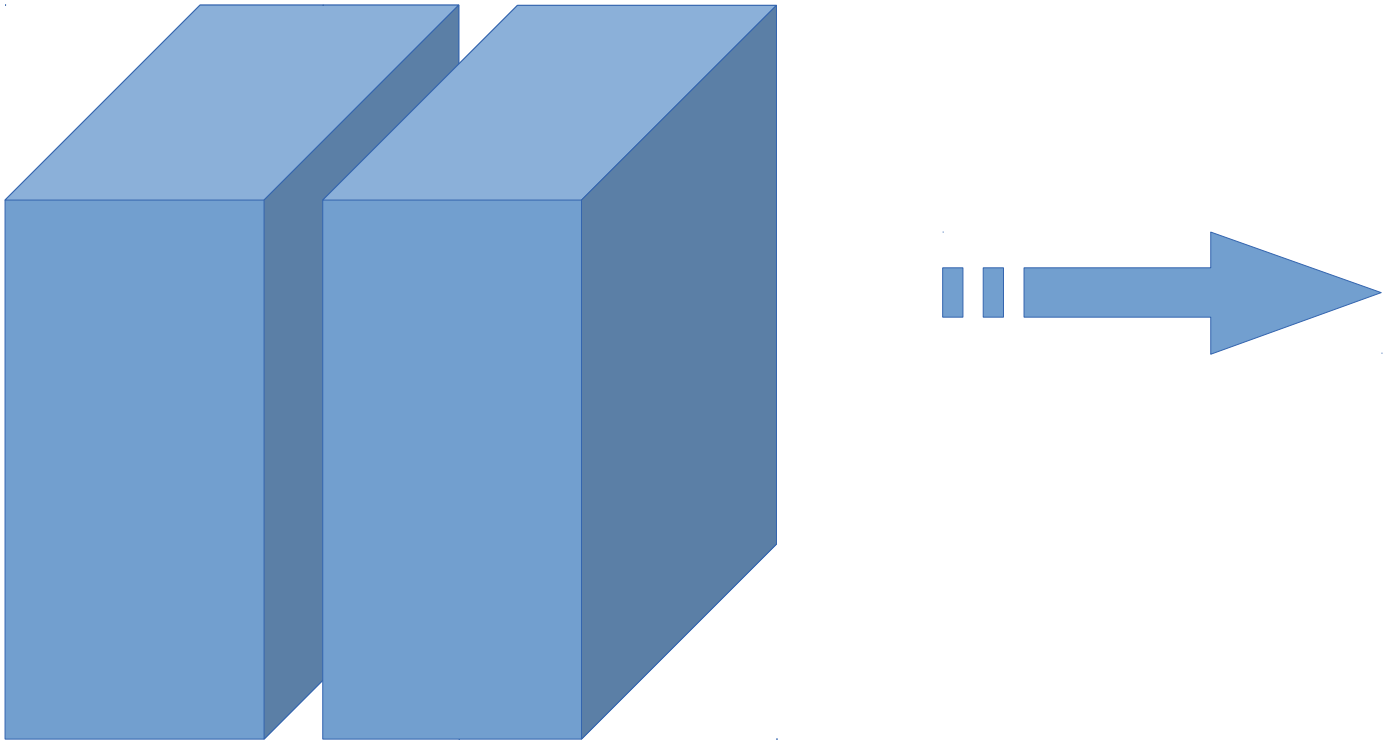
Escalar ZODB

O mediante escalamiento horizontal mediante soluciones como RelStorage, zrs, neoppod



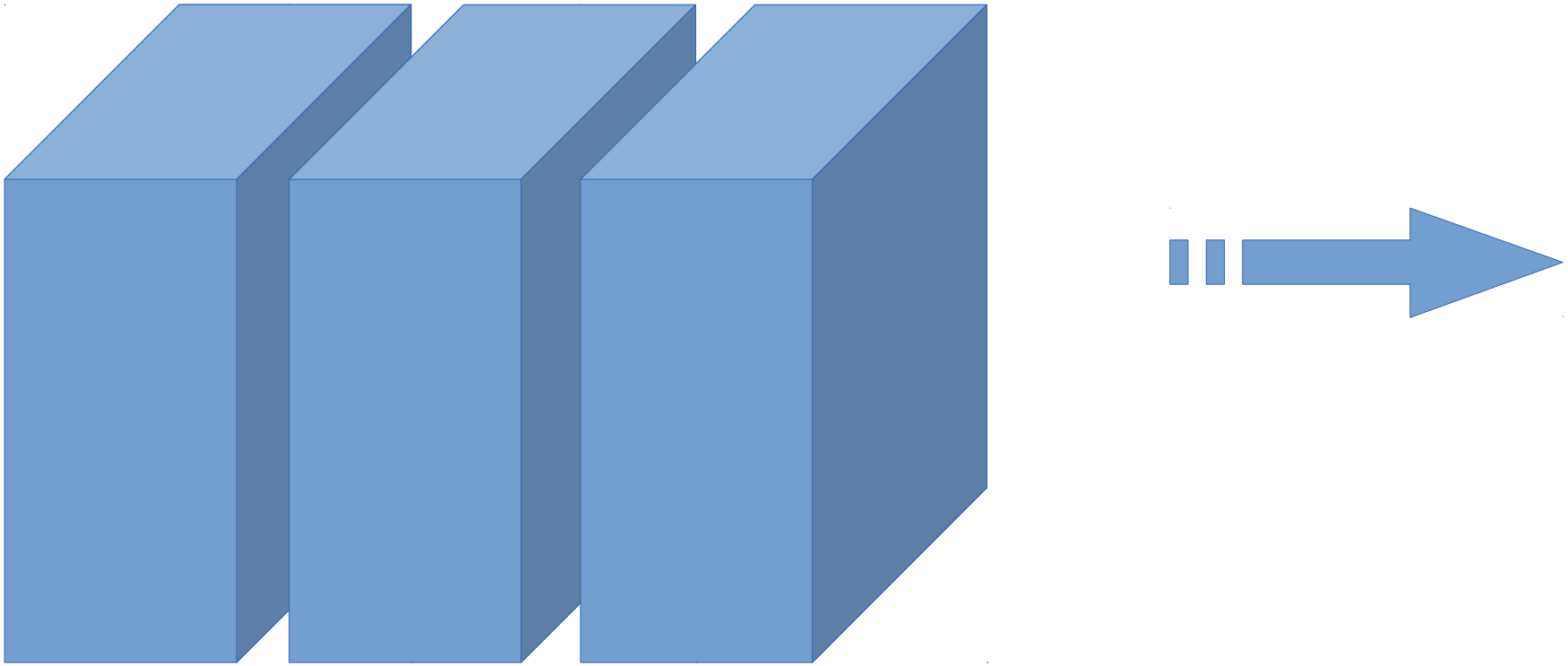
Escalar ZODB

O mediante escalamiento horizontal mediante soluciones como RelStorage, zrs, neoppod



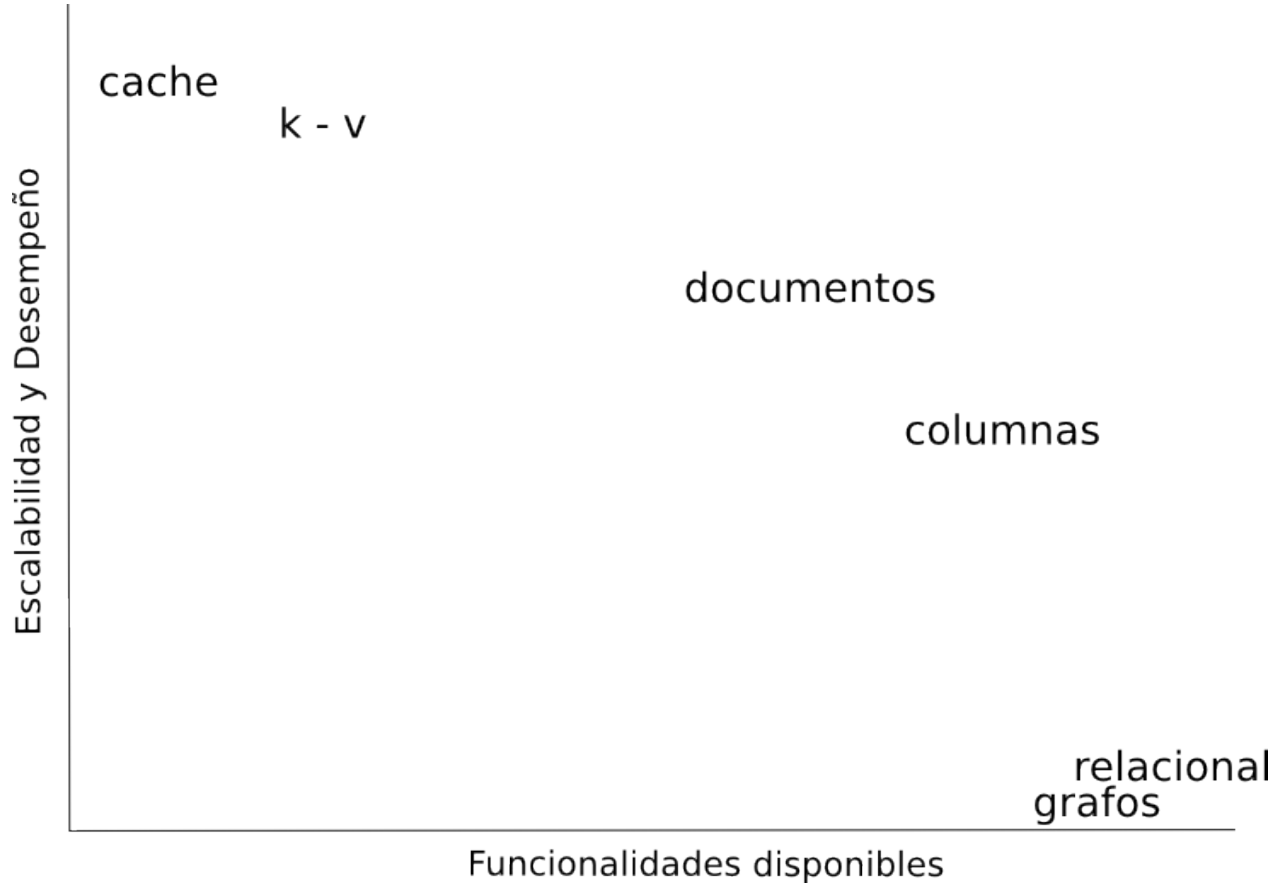
Escalar ZODB

O mediante escalamiento horizontal mediante soluciones como RelStorage, zrs, neoppod



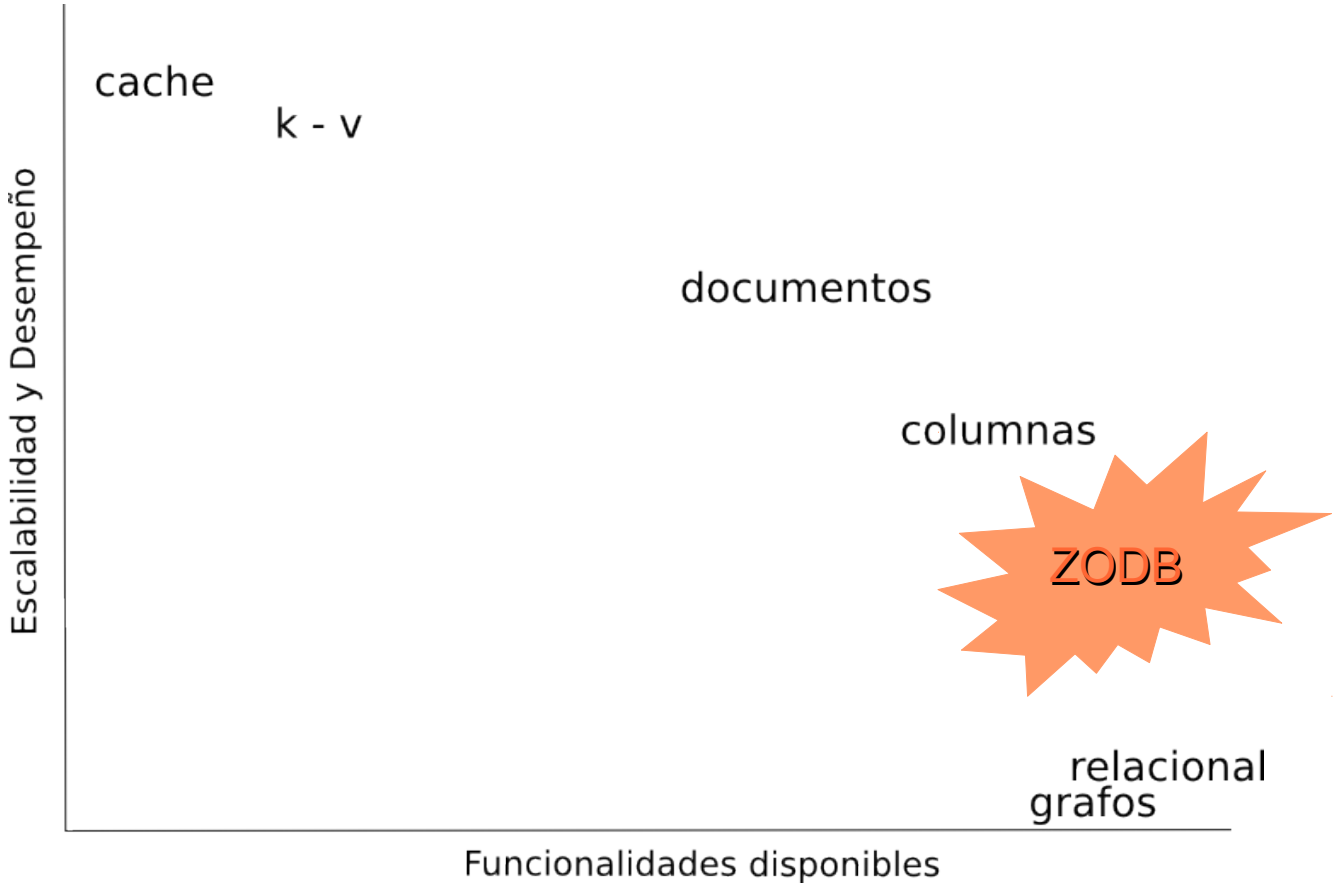
Palabras Finales

En el mapa de las Bases de Datos Modernas podemos ver lo siguiente



Palabras Finales

En el mapa de las Bases de Datos Modernas podemos ver lo siguiente



Palabras Finales

- ZODB es la forma más sencilla de persistir datos en Python.
- Puede ser usada como una base de datos de llave valor, de documentos o de objetos, todo dependerá de como se programe la solución
- ZODB es simple y divertido, tan simple que sorprende.
- Para información adicional visite:
  http://atmantree.com/go/2013/07/breve-introduccion-a-zodb/ (blog post)
  http://www.zodb.org (página oficial)

Gracias por su Tiempo

Estoy a la orden en

@atmantree

TuBaseDeDatosLibre.org