



introduction to a web object publishing framework

Rik Hoekstra

What is Zope

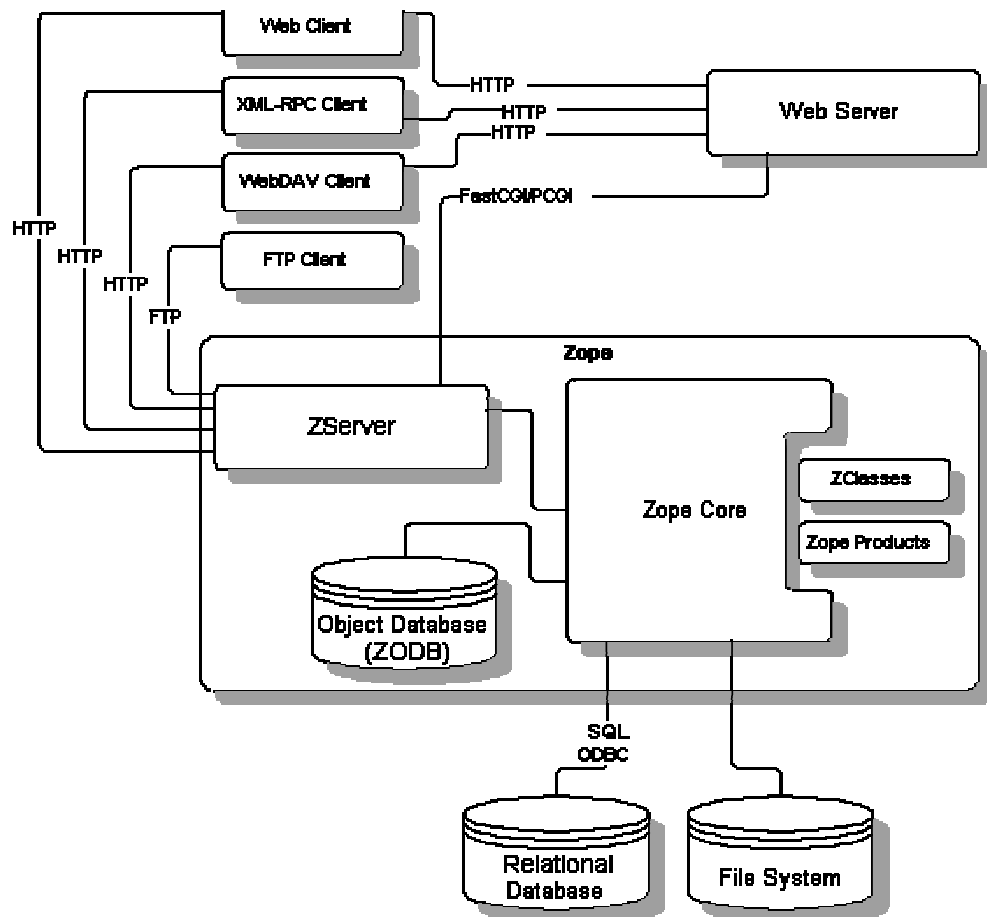
- Web Application Framework
- Open Source
- Through the Web (but not only through the Web)
- Free, Open, Adjustable
- Extensible, Modular, Cross Platform
(Linux, Unix, WinNT...)
- **Zope is flexibility**

Background

- Spinoff from newspaper company
- 2 ancestors: Principia and Bobo
- Open Source after pressure from investor
- Digital Creations made Zope, but
“Zope must be bigger than DC”
- Changing fast 1998: version 0.9
- Sept 2000: version 2.2.2

Zope communities

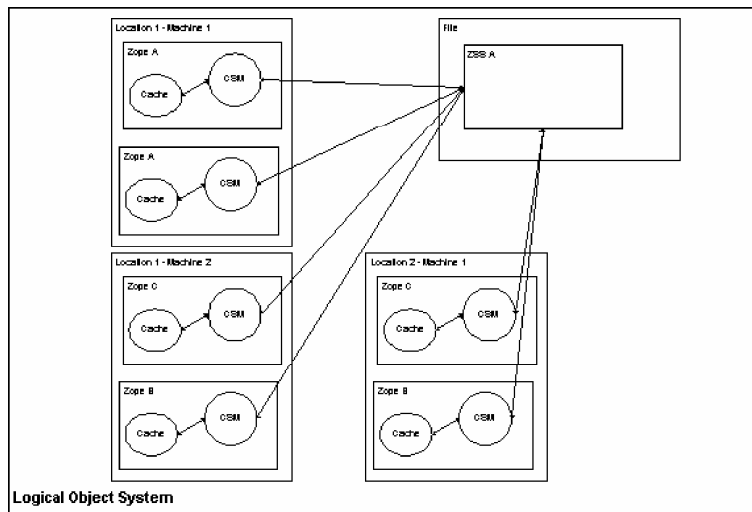
- Growing from the start
- Very active community
100+ maillist messages/day
220+ Products.
- Many local communities
- some figures:
maillists: 2500 members (main lists)
site visits: 220 000+/month (ca. 2 M page views)
- How big is Zope – unknown



Framework

- Modularity = flexibility
 - db connectivity (Oracle, Sybase, ODBC, MySQL, Postgres, LDAP.....). They all return search objects
 - storage: ZODB, but also alternative storages (dbm, directory storage, rdbms, ZEO)

ZEO diagram



Programming

- DTML – template language (but too powerful)
- ZSQL Methods program databases
- Python external methods (filesystem)
- Python Methods (database)
- other programming options coming (Perl Methods, XSL Methods)

Zope – Objects

- Everything is an object:
 - layout
 - logic
 - containers
 - search objects
- Separation of layout/code/content

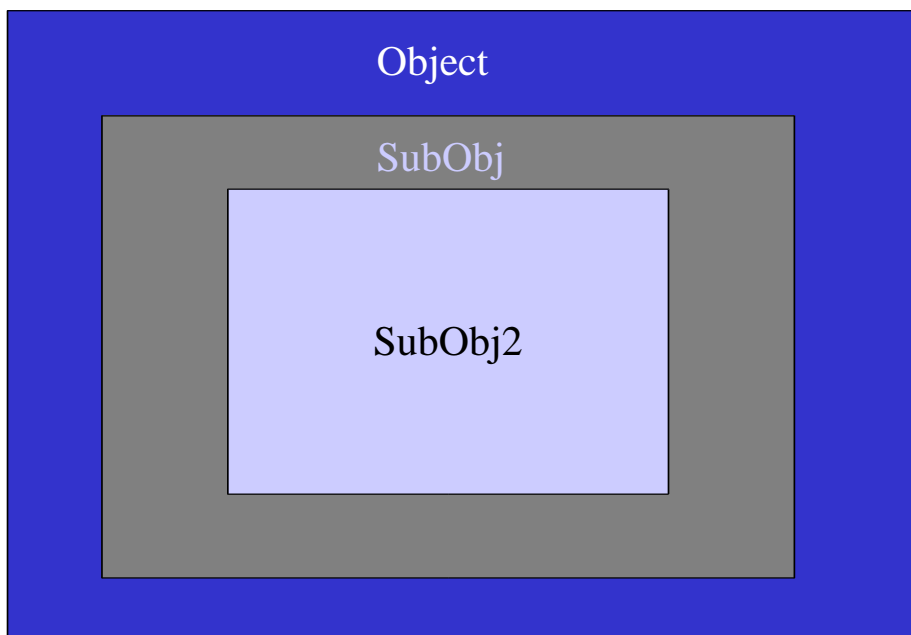
Objects

- inheritance
- acquisition (dynamic inheritance)
- methods
- properties
- New object types definition:
 - filesystem (Python)
 - Web (Zclasses)
(and combinations)

Publishing

- Zope publisher is a component
- Current publishing modes:
 - HTTP
 - WebDAV
 - FTP
 - XML-RPC (future: SOAP?)
- Web publishing: Objects to URLs

Example: Objecthierarchy -> URL



Resulting URL:

<http://someserver/Object/SubObj/SubObj2>

Zope Security

- Through the Web:
 - security delegation
 - fine grained (per object)
 - organized by:
 - users
 - roles
 - permissions

Zope security (2)

- Types of objects have properties/actions for which permissions are needed (add, edit, delete, change properties, add subobjects etc).
- Users get assigned roles (Manager, Layout Manager, Content Manger etc)
- Each permission gets assigned to roles -> users with roles have permission

	Permission	Roles		
<u>Acquire permission settings?</u>		<u>Anonymous</u>	<u>Manager</u>	<u>Owner</u>
<input checked="" type="checkbox"/>	<u>Access contents information</u>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<u>Change permissions</u>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<u>Delete objects</u>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<u>Manage properties</u>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<u>Manage users</u>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<u>Undo changes</u>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<u>View</u>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<u>View management screens</u>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Future

- Tighter XML integration (Zope as an XML Document, Xpath access, XSL Methods)
- DTML reworking (?)

Zope – Balance

PRO

- innovative
- easy installing, scalable, platform independent
- rapid development
- community
- Open Source
- flexibility

CONTRA

- documentation still weak
- not tightly integrated development modes →
- hard choosing programming modes