Scaling Plone across UNC School of Medicine

Ian Anderson and
Andy Leeb

Who we are

Andy Leeb
ableeb@med.unc.edu

Ian Anderson
ianderso@med.unc.edu

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

Where we are now

- ~150 sites across ~40 containers
- ~3500 content editors
  - smaller number of heavy users
- ~20,000 unique visitors/day

Inception

- New Dean of Medical School/CEO of hospital requested a consistent look and brand for all UNC Health Care properties.
- Web Steering Committee was formed to direct the project

Reasons for CMS

- Plone suggested by previous dept manager
- Plone 2.1. site used for Curriculum site
  - Curriculum app - first and second year curriculum materials, class times, schedules...
- Or should we use a set of Contribute templates?

Requirements

- Members of web steering committee defined requirements
- Based on the Dean’s request for consistent look and feel
- Requirements also from interpretation of “user” requests
    - must have banners! With Flash! Rotating! Random! Yay!

Transitions

- Transition from disparate environments to unified look and feel
  - no consistent branding, people were “lost”
  - people used to Dreamweaver/Contribute
  - custom web apps

Systems Architecture

- Solaris
- Sun Fire Hardware
    - 1U servers with 4 drives
    - had hardware RAID, but not used
    - overheating
- Open Solaris packages out of date or non-existent
- Limited expansion/migration ability
- Fixed resources

Systems Architecture II

- VMWare
- Linux
  - Fedora 11-18
- Apache
  - used as a redirect engine, and to serve old static sites
- Squid
  - Reverse Proxy Cache
- HAProxy
  - load balancer
- Several Zope containers per "machine"

System Architecture III

- Hostout
    - perform buildouts on many machines from central location
    - each zope container is represented by a hostout host
- Issues
    - HAProxy introducing disconnects
    - Squid seems to be introducing too much overhead
        - purging, by timeout but not by request
    - Apache configuration is quite long

Plone 2.5 and the Monolithic Stack

- Scaling - Apache, Squid, CacheFu
  - Difficult with only one container
- Deployments
  - before buildout
  - early buildout
- trac/svn - professional development tools vs Contribute
- Learning about eggs, deployment
- Custom products/early GenericSetup
- Performance metrics

Plone 3

- Plone 3 product architecture
  - central services consumed - ID Mgmnt (LDAP/onyen), transition from SOM ID, decision to not use Shibboleth, VMWare infrastructure
  - Google Search Appliance
  - changes in product layout
  - changes in namespace
  - custom products (subsite)
- Plone 3 system architecture
  - moving from one zope container to many
    - Metrics

Plone 3 II

- Custom migration code
- Unit testing, svn post-commit hooks, ci
  - buildbot, hudson
- pypi mirror - pypi.med.unc.edu
- Custom site for news.unchealthcare.org
  - more specialized
  - “all day app”
  - showcase Plone’s abilities
  - features that end up in the base products
  - http://news.unchealthcare.org

Migration - Plone 4

- Python 2.6
- Manual migration
    - migration steps
        - product migration
            - GSA -> Google Site Search/Google Custom Search
            - move away from custom in-house products
            - p4a.calendar anyone
            - no more flash!
    - strip away legacy decisions
- Transmogrifier for stubborn sites
- Performance tweaks

Migration - theme

- Migration details
    - objectives
    - customer
- Automating the theme migration with Jenkins
    - shell script restore
    - stand up site
    - run migration steps
        - automated screen shots
        - allow customer to finalize
    - deploy
- Selenium

Future Steps

- Fedora 18
  - OpenShift anyone?
  - system python
  - system varnish, shallower stack
- Further automation with Selenium
- Plone 4.3

Conclusion

- Encapsulate customizations in your products
- Thoroughly vet 3rd party addons
  - collective.gsa/collective.indexing
- Diazo is a win
- Automate migrations as much as possible
- Use system packages

Questions