Building ArsDigita Portals #2
Frank Bergmann <fbergmann@competitiveness.com>
Barcelona, March 1st, 2001
Content

- Recap: ACS Architecture
  - Reliable Web Services
  - AOLServer
  - AOLServer against Apache
  - ACS Application Architecture

- How To Build Your Portal
  - Define the Project
  - Setup an ACS Server
  - Build a First ACS Portal
  - What Went Wrong?
  - Form Your Community
  - Build Custom Modules
  - Make Money
Recap: ACS Architecture

- Reliable Web Services
- AOL Server
- AOL Server against Apache
- Connection Pooling
- ACS Application Architecture
Reliable Web Services: Problems

- Behaviour under heavy load
  - Trashing
  - Memory overflow
  - Infinitely growing queues
- Thread programming
  - Forgot to lock critical regions
  - Deadlocks
  - Too many/too few threads
- Continuus running processes
  - Memory leeks
  - Maintenance at runtime

http://www.arsdigita.com/asj/arsdigita-server-architecture
Reliable Web Services

**Philip Greenspun:**
- „Leave the hard stuff of concurrency control and transaction atomicity to a standard relational database management system (RDBMS)“
- „Develop pages in a safe interpreted language“

=>
- It’s good to know about threads, but the more you know, the more you rely on working solutions
- Systems under heavy load behave very differently. For example, WindowsNT & IIS still run out of memory under heavy load

http://www.arsdigita.com/asj/arsdigita-server-architecture
Traditional CGI architecture.
20 requests per second for database-backed pages = 40 new programs started per second.

AOLserver architecture.
Database connection-pooling: 20 requests per second for database-backed pages = 0 new programs started per second
AOLServer against Apache

**Apache**
- Maintained by Apache Group
- Modular
- Feature rich: Virtual Servers, fancy authentication, ...
- DB driver part of CGI program

**AOLServer**
- Maintained by AOL
- Monolithic
- Designed for one purpose: being fast
- DB driver part of server

http://www.arsdigita.com/asj/aolserver/introduction-1.html
Connection Pooling

- Normally, Oracle spawns a new thread/process for each incoming connection.
- Connection setup is slow.
- A limited number of connections reduces the maximum workload.
ArsDigita Application: Context

WWW Clients

Internet

WWW Server

Application Code

Oracle Database

Internet Server
Modules consist of:
- TCL code for dynamic pages
- SQL code for DB queries
- SQL Code to create data model

=> See 2\textsuperscript{nd} part of the talk
Disadvantages

- Disadvantages:
  - It’s not Java
  - Not very well suited to deal with XML
  - Not very well suited to deal with complex business logic.

http://www.arsdigita.com/asj/arsdigita-server-architecture
Summary

- ACS Architecture designed for being:
  - on the Web
  - fast
  - reliable
  - easy to learn/program

- But:
  - Oracle 8i and Linux need a SysAdmin and
  - I would program a math problem in Java/C++

http://www.arsdigita.com/asj/arsdigita-server-architecture
5. How To Build Your Portal?

How to Build Your Portal?

- Define the Project
- Setup an ACS Server
- Build a First ACS Portal
- What Went Wrong?
- Form Your Community
- Build Custom Modules
- Make Money
Define the Project

Driving School Portal
Case Study

- Idea
- Target Group
- Why Join the Portal?
- Additional Contents
- Which ACS Modules?
- Marketing
- Make Money
Setup an ACS Server

What to do?

1. Learn some Linux
2. Learn TCL
3. Learn SQL
4. Get a Linux server
5. Install ArsDigita
6. Install Oracle

How to do?

- Install Linux at home
- ACS problem set 1
- ACS problem sets 1 & 2
- PC with 128MByte RAM
- Read online doku
- Read online doku

Get together with some friends who have done it already
Build a First ACS Portal

1. Get an idea of what you want to build
2. Define a web design
3. Configure some existing modules
4. Make some small changes to the modules
5. See why nobody is using your portal
6. GOTO 1
   or continue with next slide
What Went Wrong?

- “Nobody likes to enter an empty bar” effect:
  - Create artificial “noise”
  - Ask your friends to participate
  - Actively form your community

- Application modules doesn’t 100% fit your needs:
  - Analyze in detail user behavior (ask your girlfriend/boyfriend)
  - Build custom modules
# Form Your Community

## What to do?

1. Setup an initial community
2. Make people stay in your portal
3. Attract/maintain users

## How to do?

- Tell your friends to participate
- Get killer content
- Design apps for people to stay
- Make “Strategic Partnerships”
- Import contents from other sites
- Assure high quality/usability
- Remove old/bad contents
Build Custom Modules

1. Get an idea of what you want to build
2. Define a web design
3. Make a “Wemo” (=Workflow Demo) for new modules
   - Present the Wemo to friends & family.
   - The Wemo will save a lot of time during development.
4. Configure some exiting modules
5. Make an interaction model
6. Make a data model.
7. Write the TCL pages
8. Test the system together with some friends
9. See why nobody is using your portal
10. GOTO 1
Make Money

- Making money with a portal today is nearly impossible.
- You can try to sell your portals to people who still believe they can make money...
6. Related Literature

- Ars Digita: http://www.arsdigita.com/
- TCCG: http://www.competitiveness.com/
- ACS Documentation: http://www.arsdigita.com/doc/
- The Online Bible: http://www.arsdigita.com/books/panda/