SSI = Sieciowe Systemy Informacyjne

Wykład nr 4

Serwery WWW (rząd 2, app web).

- - serwer www zawiera strony html statyczne lub generowane dynamicznie (np. przez skrypty php)
  - *jak to działa*: przeglądarka wysyła żadanie http do serwera www, skrypt php tworzy html (z CSS i z JS), html jest odsyłany w odpowiedzi http do przeglądarki, przeglądarka pokazuje html, JS w html działa w przeglądarce...
  - rozróżniać kod rz.1 (JS w przeglądarce) i rz.2 (skrypt php na serw. www)

- CGI, skrypty CGI
  zewnętrzny program/skrypt uruchamiany przy KAŻDYM żądaniu http
  otrzymuje dane "żądania http" przez zmienne środowiska i stdin
  zwraca odpowiedź http przez stdout
  konfiguracja: zależna od serwera www, katalog cgi-bin, ...
  php może działać w tym trybie na KAŻDYM serwerze www (php-cgi)

```
#!ścieżka_do_tclsh
  # specjalna pierwsza linia skryptu, chmod u+x skrypt, tclsh to interp tcl-a
lappend auto_path ścieżka_do_tcllib; package re ncgi
  # ncgi to pakiet obsługi CGI w j. Tcl
ncgi::parse
ncgi::header "text/html; charset=iso8859-2"
puts "<p>Witaj drogi kliencie, zmienna=[ncgi::value zmienna]</p>"
  # spodziwam się "zmienna" w parametrach żądania http...
```

- apache
  tworzy pulę procesów...
  mod_php, php/CGI, php/FastCGI
  FastCGI to lepsza wersja CGI; utrzymuje się pulę procesów...

- nginx
  php/CGI, php/FastCGI

- AOLserver
  tworzy pulę wątków... wątki vs procesy ?!
  zarządzanie pulą wątków: max/minthreads, inicjalizowanie wątków, "blueprint"
  pliki .adp i .tcl, kod rezydentny .tcl w modułach(!), zmienne nsv
  AOLser API: komendy ns_*, np. ns_conn, ns_queryget, ...
  framework webowy nad AOLser: OpenACS, dotLRN, ProjectOpen
  patrz: http://mhanckow.students.wmi.amu.edu.pl/pzr420/pzr420_cw_e.htm

- AOLserver + MapServer
  //// jak uruchamiac program MapServer pod AOLser ?
  1. CGI, technika wbudowana w MapSer, sekcja WEB w pliku .map
  2. zewnętrzny skrypt Tcl używający pakietu MapscriptTcl1.1; narzędzie SWIG
  3. MapSer + MapscriptTcl1.1 w wątkach AOLser (???)

  keszowanie plików z obrazkami, jak budować nazwę pliku ???
  pokazać przykład z AOLser, pokazać wyciek pamięci SWIG

- servlety i JSP

  język Java, "kontener serwletów" Tomcat (serwer www z obsługą serletów i JSP), oficjalna specyfikacja Servletów i JSP,
  część architektury J2EE (4 rzędowej, rzęty wewnętrzne to "middleware")

  servlet: klasa Javowa dziedzicząca z klasy javax.servlet.http.HttpServlet, metody doGet, doPost odpowiadające na żądania http GET, POST;
  serwlety posiadają obiekty: request, response, session, context (= aplikacja serwletowa = prefix url-a),
  atrybuty sesji (=zmienne sesyjne), atrybuty aplikacji, atrybuty request (przy forwardowaniu z serwletu do JSP)
  jeden serwlet obsługuje jeden url lub wiele - kwestia konfiguracji Tomcata...

  JSP: plik podobny do html, ze wstawkami <% kod java %>, <%= wyrażenie java%>, z tagami <jsp:???>, tłumaczone automatycznie do servletu;
  można też używać taglibs (dodatkowych bibl. z tagami, np JSTL)
  możliwość korzystania z JavaBean (<jsp:useBean>) i innych obiektów Javy...

  pytanie: dlaczego jsp podobne do html ???

  pomysł: "skryptowe serwlety", osadzić w serwlecie javową wersję interpretara j. Tcl, Python itp... pokazać przykład

Serwery WWW/ frameworki webowe

- frameworki webowe ułatwiają tworzenie aplikacji webowych,
  np. pisanie kodu skryptów PHP (organizacja kodu)
- *naczelna zasada*: oddzielać "prezentacje" (tworzenie html)
  od "logiki biznesowej" (wyciaganie danych i ich przetwarzenie)
- wzorzec architektoniczny **MVC** (ang. Model View Controller)
  Model = tabele w bazie danych, kod operujący na danych z bazy
  View = kod tworzący html pokazujący dane
  Controller = w app web: obsługa url-a, jest częścią frameworka webowego
- przykłady frameworków webowych

  Symfony: j. PHP, framework podstawowy, tj app web programujemy od zera ...

  Drupal: j. PHP, framework CMS-owy, CMS (ang. Content Management System), gotowa aplikacja zorientowana na zarządzanie "contentem", którą się jedynie nieco przystosowuje do klienta ...

  OpenACS: j. Tcl, serwer AOLserver, framework podstawowy + CMS-owy, duże API, ciekawa aplikacja XOWiki, zorientowany na "portale społecznościowe", używany często jako platforma e-learning,

  aplikacje RIA: 1 strona www z dużą ilością JS

- Symfony (v1.?)

  struktura katalogów projektu sf:

```
./apps
./apps/app1
./apps/app1/templates
./apps/app1/lib
./apps/app1/config
./apps/app1/modules
./apps/app1/modules/mod1
./apps/app1/modules/mod1/actions
./apps/app1/modules/mod1/templates
./apps/app1/i18n
./cache
./lib
./plugins
./data
./config
./log
./web
./web/images
./web/js
./web/css
```

  – programy narzędziowe sf, np. tworzące strukturę katalogów projektu
  – aplikacja "app1" dzieli się na moduły ("mod1", ...)
  – w każdym module mamy actions i templates:

```
// w pliku mod1/actions/actions.class.php
class mod1Actions extends sfActions
{  public function executeShow()
   {  $today = getdate();
       $this->hour = $today['hours'];
       $this->a1 = 111;
       $this->a2 = 222;
   }
}
// w pliku mod1/templates/showSuccess.php
<p>Tra la la !!!</p>
<p>hour=<?=$hour?></p>
<p>a1=<?=$a1?></p>
<p>a2=<?php echo $a2?></p>
<p>Tra la la !!!</p>
```

- Symfony (v1.?)

  Co wspiera Symfony???

  - obsługa sesji i zmiennych sesyjnych,
  - formularze HTML,
    także automatyczne ich budowanie na podstawie definicji tabel w bazie!
  - operacje na bazach danych, ORM (ang. Object Relational Mapping)
  - funkcje pomocnicze w template-sach, np. fun. to_link()
  - programy tworzące automatycznie: strukture katalogów projektu, pliki konfiguracyjne YAML, ...
  - keszowanie (katalog cache)
  - templates master/slave,
    w mod1/templates podajemy maly template (slave), który zostanie osadzony w wiekszym (master) zdef. w app1/templates

- OpenACS (v5.6)
  - ogólna organizacja: drzewo portali i pod-portali (subsites), w portalach montuje się instancje aplikacji ...
  - wbudowana obsługa użytkowników, grupy użytkowników, każdy subsite ma grupe subsite-ową użytkowników, możliwość rejestrowania się użytkowników itp; *innymi słowy wiele gotowych/ użytecznych modeli danych (tabelek)*
  - rozszerzalność przez pakiety, dwa typy pakietów: aplikacje i serwisy, aplikacje przystosowane do wielokrotnych instancji, dużo gotowych aplikacji (xowiki, forum, file-storage, calendar, photo-album itp)
  - przykład aplikacji: XOWiki, zbiór stron wiki,
    hipertekst (linki), dokumenty złożone (includelety)
    xowiki::Page - zwykła strona, przechowywane w elem. CR, obiekt klasy xowiki::Page
    xowiki::Object - strona programowalna, proc content zwraca content
    xowiki::Form i xowiki::FormPage - pusty i wypełniony formularz, możliwość gromadzenia danych przez formularze
  - przykład serwisu: CR (ang. Content Repository), CR zawiera "elementy CR" w "folderach", dla elementów przechowuje się ich "wersje"
  - obiekty acs: są to trwałe "obiekty bazodanowe" (klasa - tabela, obiekt - wiersz tabeli), aplikcje przechowują dane w obiektach acs, można definiować prawa użytkowników/grup do ob. acs, wiele std. informacji jest przechowywanych przy ob. acs

- struktura pakietu oacs: aplikacja "qqq1", dwa podkatalogi:
  qqq1/tcl - rezydentny kod tcl, wysokopoziomowe API pakietu qqq1 (!)
  qqq1/www - templatesy oacs, złożone z 2 plikow strona.tcl i strona.adp
  templates-y master/slave, tagi <master> i <slave>
  przy tworzeniu templates jest wsparcie dla formularzy (ad_form) i list (template::list)
- OpenACS API: zbudowane nad AOLser API, komendy ad_* i inne, historyczne warstwy api: pgplsql, tcl api (stare i nowe), xotcl api (xotcl - obiektowe rozszerzenie tcl-a), każdy zainstalowany pakiet rozszerza to API...
- patrz: http://openacs.org

- Drupal
  - CMS = Content Management System (system zarządzania treścią)
    wprowadza się dokumenty/artykuły/??? przez przeglądarkę
    (od strony klienta!! inaczej niż przy statycznych stronach www...)
  - duże menu po lewej stronie, po prawej bieżący dokument,
    w dokumentach mogą być linki do innych dokumentów
  - Drupal to "framework CMS-owy" ...
    można go rozszerzać, np. o nowe typy dokumentów,
    można modyfikować menu, dodawać "bloki", ...
    rozszerzanie przez "moduły" z "hakami" (funkcje PHP o spec. nazwach)
  - Drupal jest zaprogramowany w PHP,
    może używać różnych baz danych np Postgres...
    patrz: http://drupal.org
  - zalety: od razu mamy gotową aplikację,
    wady: nie zawsze łatwo przystosować do potrzeb

- RIA = Rich Internet Application
  - główna idea frameworków webowych RIA :
    dużo kodu JS (JavaScript) w przeglądarce, często w 1 stronie www
    ... skąd się bierze ten kod ???
  - **Vaadin** - aplikację pisze się jak aplikację desktopową w j. Java,
    wszystko jest automatycznie tłumaczone do JS i wędruje do przeglądarki,
    zdarzenia są odsyłane do serwera i obsługiwane przez kod w Javie,
    można używać wyrafinowanych komponentów wizualnych (GWT),
    patrz: http://vaadin.org
  - **Echo2/3** - podobna idea, starsze, nakładka Tcl o nazwie **Aejaks** ...
    patrz: http://echo.nextapp.com/
    http://aejaks.sourceforge.net/
  - zalety: łatwość programowania (jak app desktopowe w Javie: komponenty, kontenery, obsługa zdarzeń itp)
    wady: nie skaluje się !!! nie działa na tabletach ???