Tools and Technologies: COMA (conference management system)

Alexander Derenbach

November 15, 2004

1 Introduction

This is a description of the tools that should be used to implement COMA. Concept and specification are covered by other papers.

2 The Architecture

COMA should be implemented as a client/server architecture and be accessible over the Internet. On account of this it is consequent to use an available server/client architecture such as webserver/webbrowser. The following section will discuss some of the technologies and software products in this field.

3 The Tools

3.1 The Webserver

The available webservers can be classified into 'free' and 'commercial' products. Commercial, that is you have to pay for them, are:

- Netscape Enterprise Server (commercial)
- IPlanet (commercial)
- MS IIS (commercial, only MS Server / Windows Platform)

The free webservers are:

- Apache + modules (free)
- Roxen (free)

Out of these we choose Apache as our target server for these reasons:

- It's free and under GPL (GNU Public License).
- It's a good and fast webserver.
- There exists already some know-how within the team.
- It supports almost all common concepts by modules or add-ons (Java, PHP, Perl, etc.)

3.2 The DBMS (Database Management System)

Available DBMS are:

- MySQL 4.1
- PostgreSQL
- Orcale
- DB2
- Informix

Out of these we choose MySQL 4.1 as our target DBMS for these reasons:

- It's free and under GPL (GNU Public License).
- It's available on most platforms.
- There are drivers available for most languages.

Note: It has to be MySQL version 4.1 since older versions don't support SQL-Subquery syntax

3.3 The Language

Available Languages / Technologies:

- Java Pros:
  - modern object orientated language
  - platform independent
  - clear seperation of moduls possible
  - easy to automate testing the core

  Cons:
  - slower than some other technologies
  - partitial heavyweight
- PHP

  Pros:
  - relativ fast
  - platform independent
  - developed for display dynamic data on webclients

  Cons:
  - complex seperation of modules
  - less existing knowhow
  - tesing more complex

- Perl

  Pros:

  - platform independent

  Cons:

  - complex syntax
  - antiqued for webservices

- ASP

  Pros:

  - relativ fast
  - developed for display dynamic data on webclients

  Cons:

  - not not offical supportet for all platforms. Indeed a module for apache exists but not shure if it is in a stable business suitable release.

Out of these we choose Java as our target language for these reasons:

- There exists already some know-how within the team.
- clear seperation of components (core, data, view)
- the language concept suits to teamwork concepts (easy to seperate modules)
- possible to automize testing with JUnit

4 Realisation in Java.

Java Technologies:

- Java 2
- Java-Servlet
- EnterpriseJavaBeans EJB
- JSP
- JSP Tag Libraries
- MySQL Connector/J
- Jakarta Tomcat
- JBoss Application Server

Java Servlets, JSP and EJB are used for a clear seperation of functionality and view.

5 Links

Apache	http://www.apache.org
MySQL	http://www.mysql.org
J2SE	http://java.sun.com/j2se/index.jsp
J2EE	http://java.sun.com/j2ee/index.jsp
EJB	http://java.sun.com/products/ejb/
JSP	http://java.sun.com/products/jsp/index.jsp
Servlets	http://java.sun.com/products/servlet/index.jsp
JDBC:mysql	http://www.mysql.com/products/connector/j/
Jakarta Tomcat	http://jakarta.apache.org/tomcat/index.html
Jboss App. Server	http://www.jboss.org/