

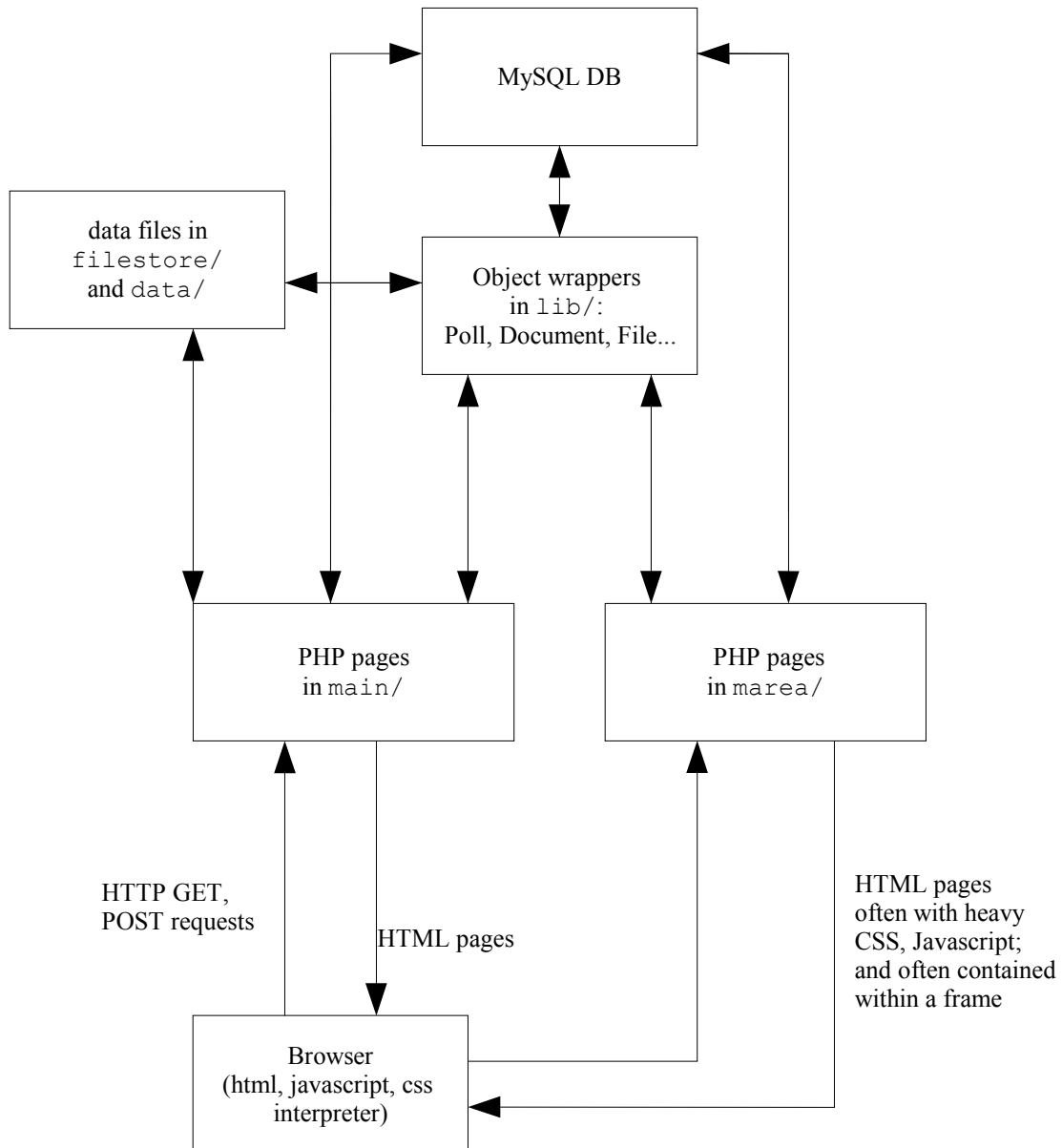
# Technical Design of POD

XXX This document is a mere start of draft. It is NOT anywhere CLOSE to complete!  
Hope it helps though. XXX

POD is programmed in PHP using a MySQL relational database as well as the filesystem to store data. It makes liberal use of Javascript, CSS, frames, and other client-side browser technologies to create a more dynamic interface than can be done in pure HTML.

## ***POD Architecture***

...isn't the cleanest ever...



One somewhat unique aspect of POD is that, especially for the meeting area, lots of UI logic exists not as PHP code, but rather as Javascript embedded through php files. The client's browser's javascript/dhtml engine actually does more UI work than the server's php engine, for the meeting area.

It design deficiency that the main/ and marea/ php pages may interact either directly with the database (by executing SQL queries) or else indirectly through the object wrapper layer. Consistency through the object wrapper layer would be preferable, but at this early stage it's often more convenient to directly execute queries from a php page. (For example, newpoll.php inserts a new poll row in the database directly, instead of creating a new Poll object than calling an store-in-database method.)

## Meeting Areas

### The 7-Frame Interface

Navigating to `POD_URL_ROOT/marea` loads `frameset.php`. The entire meeting area interface remains in the same downloaded framset the whole time. It lays out the browser window with 7 frames like this:

name= <i>banner</i> banner.php	
name= <i>foliotb</i> foliotb.php	name= <i>msglisttoolbar</i> msglisttoolbar.php
	name= <i>msglist</i> glossary name: <i>discussion index</i> msglist.php  CSS: msgs.css.php
name= <i>workspace</i> glossary name: <i>folio view</i>  workspace.php, which in turn include()'s view- <code>{index,doc,discoitem,poll}</code> .php  closepoll.php ballot-submit.php edit-preface.php  CSS: ws.css.php	name= <i>curmsgtoolbar</i> curmsgtoolbar.php
	name= <i>curmsg</i> glossary name: <i>message view</i> curmsg.php  CSS: msgs.css.php

name=*blabla* is the HTML FRAMESET NAME tag, which is also the javascript identifier for the frame. The php files listed are the possible files which may appear in that frame during use.

*banner*, *foliotb*, *msglisttoolbar* and *curmsgtoolbar* all rarely reload during use. Changes to their text are accomplished via DHTML routines. (Exception: *foliotb* must reload to get an updated navigation listing of the folio items in the meeting area)

*msglist.php*'s highlighting and jumping around, is accomplished through DHTML. However, all the message headers are written via php; thus, it must reload to show new messages. *msglist.php* controls the display in *msglisttoolbar* via javascript calls.

*curmsg.php* reloads for every new message. *curmsg.php* controls the display in

*curmsgtoolbar* via javascript calls.

*workspace* reloads for every new view, of the folio index, and of each folio item.

## **Meeting Area Popups**

Popups for the meeting area – for creating new folio items and messages – actually are framesets with one big frame that does all the actual work. The external frame exists to provide a javascript object ('opener') referring back to the original 7-frame frameset window, even after the main working frame requests new pages, wiping clean its javascript memory.

## ***Random Notes***

All php pages designed to be requested by the user, are one subdirectory below POD\_URL\_ROOT and POD\_FILE\_ROOT. This means, the universal way to get at the group root is ' . . '