SYBASE TECHWAVE SYMPOSIUM 2009

Taking It All Offline with SQL Anywhere

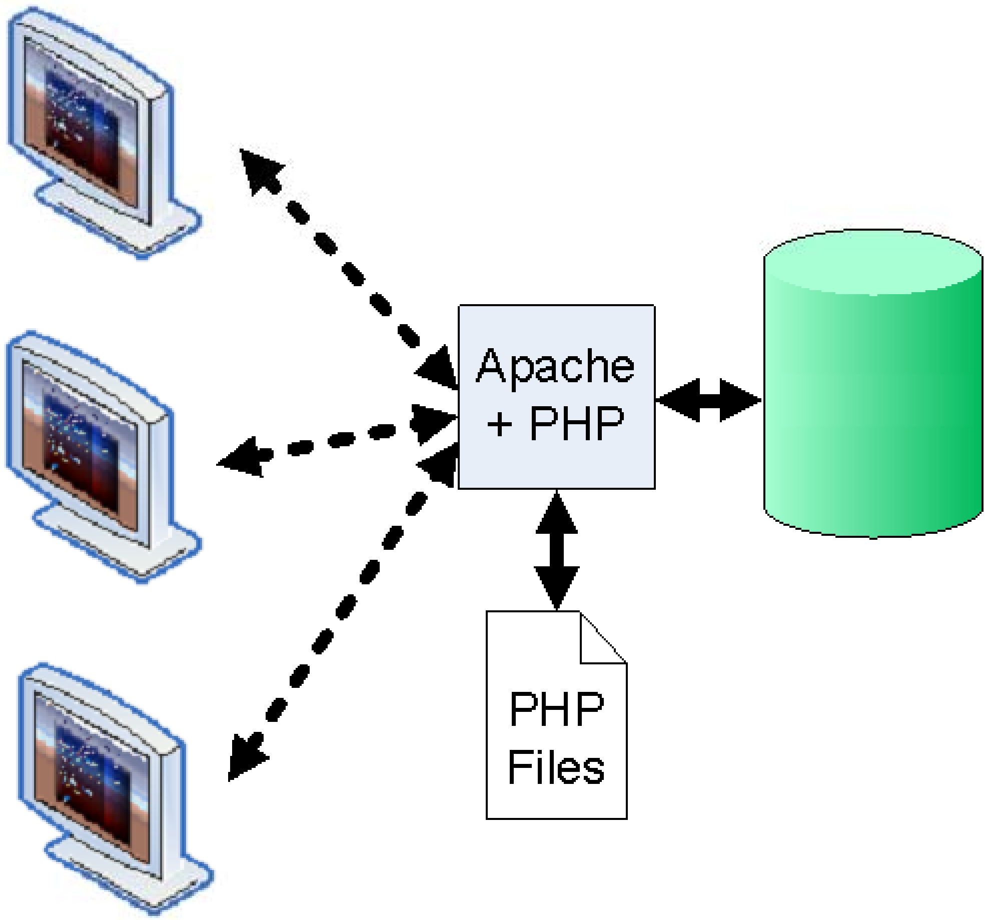
Eric Farrar

Product Manager

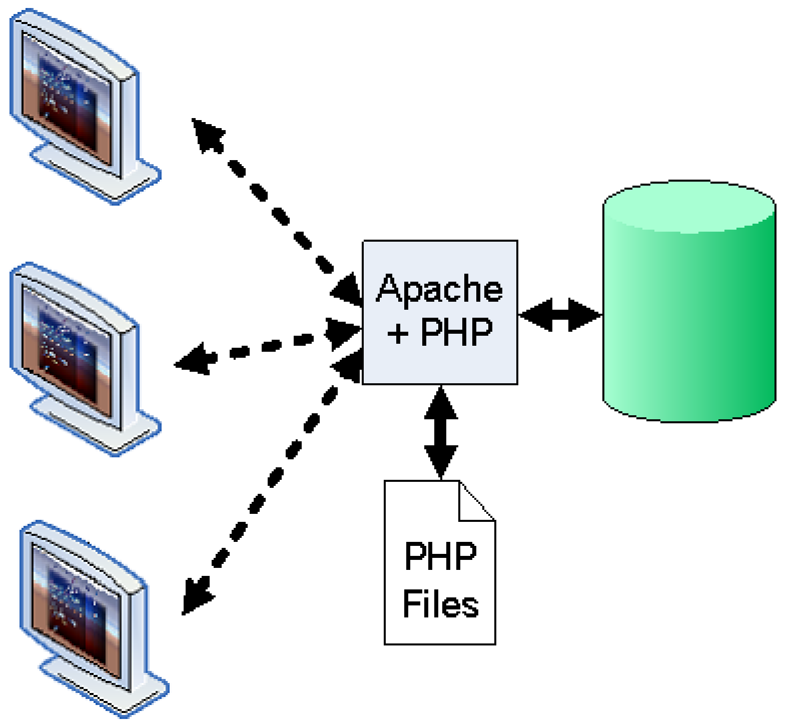
Why is Web Development Attractive?

- Zero-deployment
- No need to maintain previous versions
- Everyone updated at the same time
- Some security problems become simpler
- Web development is (usually) simpler

Simple PHP-based Application



The Problem...



How can we solve the problem?

1. Take the stack offline

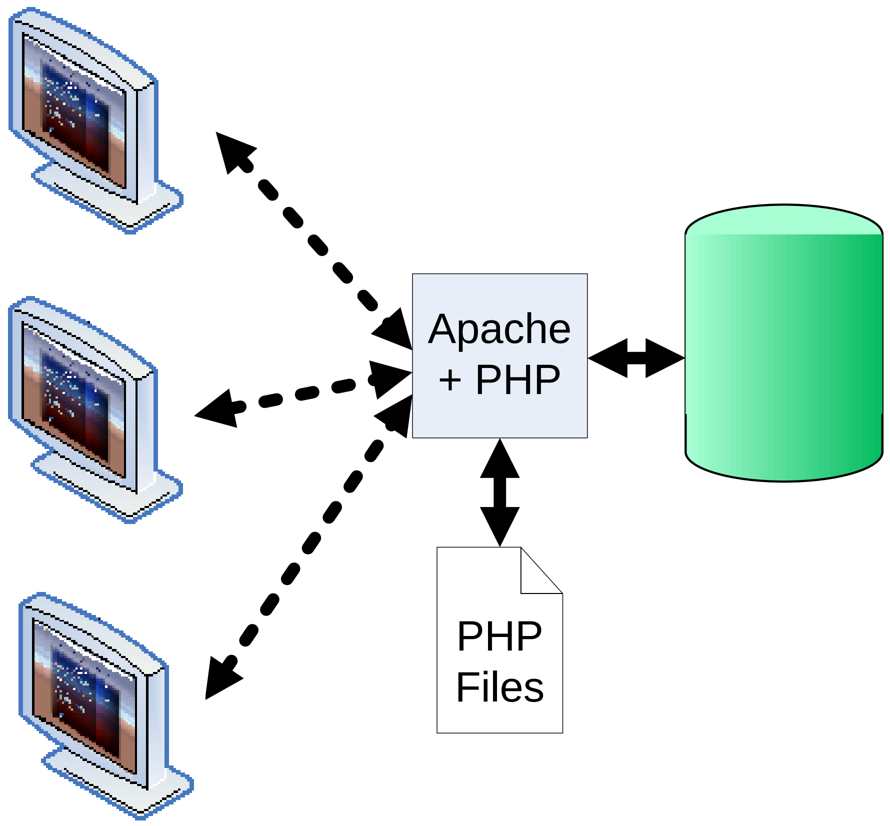
2. Rewrite the application as a client-side application

How can we solve the problem?

1. **Take the stack offline**

2. Rewrite the application as a client-side application

Sample Blog Application



Take the Stack Offline

- What do we need to host an offline version of the application?
  - Web Server
  - Database
  - PHP installation (or whatever server-side scripting language you use)
  - Synchronization

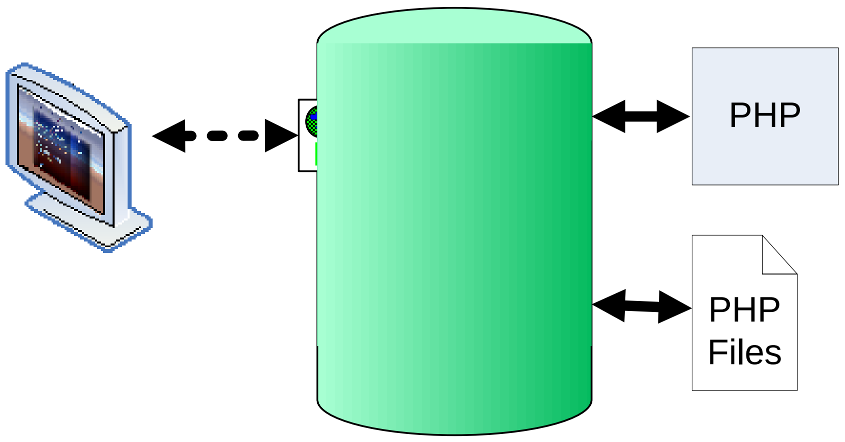
Make SQL Anywhere the Stack

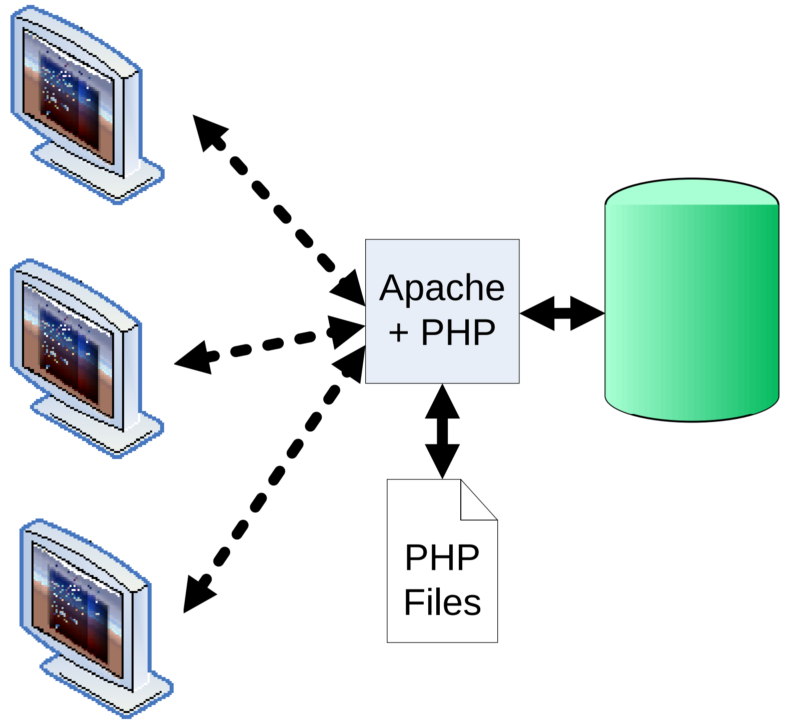
- Built-in HTTP server
  - HTTP
  - SOAP
- It *is* a database
- External Environments:
  - PHP
  - Perl
  - .NET
  - Java
  - C (external libraries)

What We Need Cont.

- Built-in synchronization
  - MySQL
  - Oracle
  - MS SQL Server
  - IBM DB2
  - SQL Anywhere
  - Sybase ASE

Stand-Alone Application

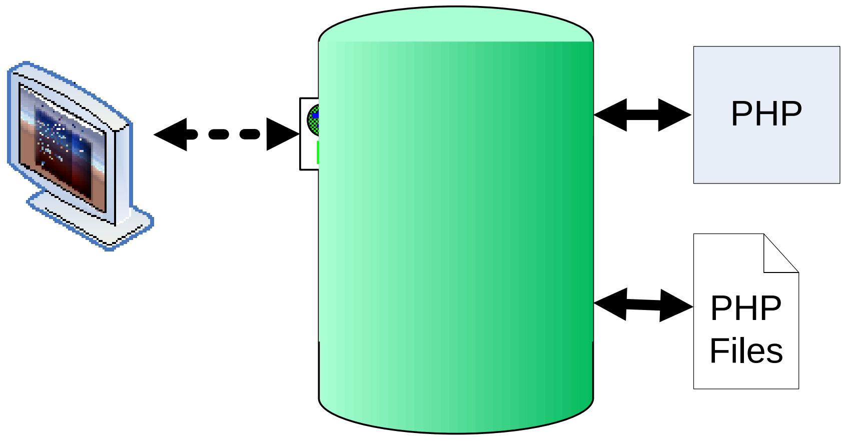


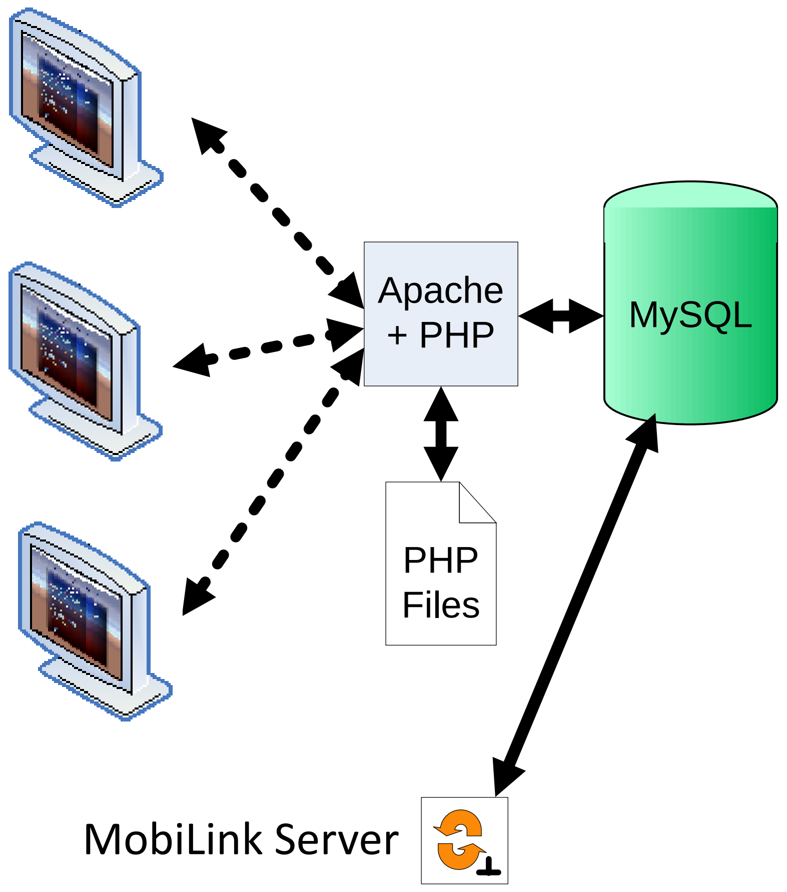


MobiLink and Synchronization

- MobiLink acts as a ‘sync server’
- Makes a regular database connection to consolidated database
- Handles all synchronization traffic
- Set up in farms to scale out (> 100,000 syncing remotes)
- Write sync scripts to handle business logic of sync
  - SQL (native dialect of consolidated database)
  - .NET
  - Java
- Conflicts detected at column or row level

Adding MobiLink

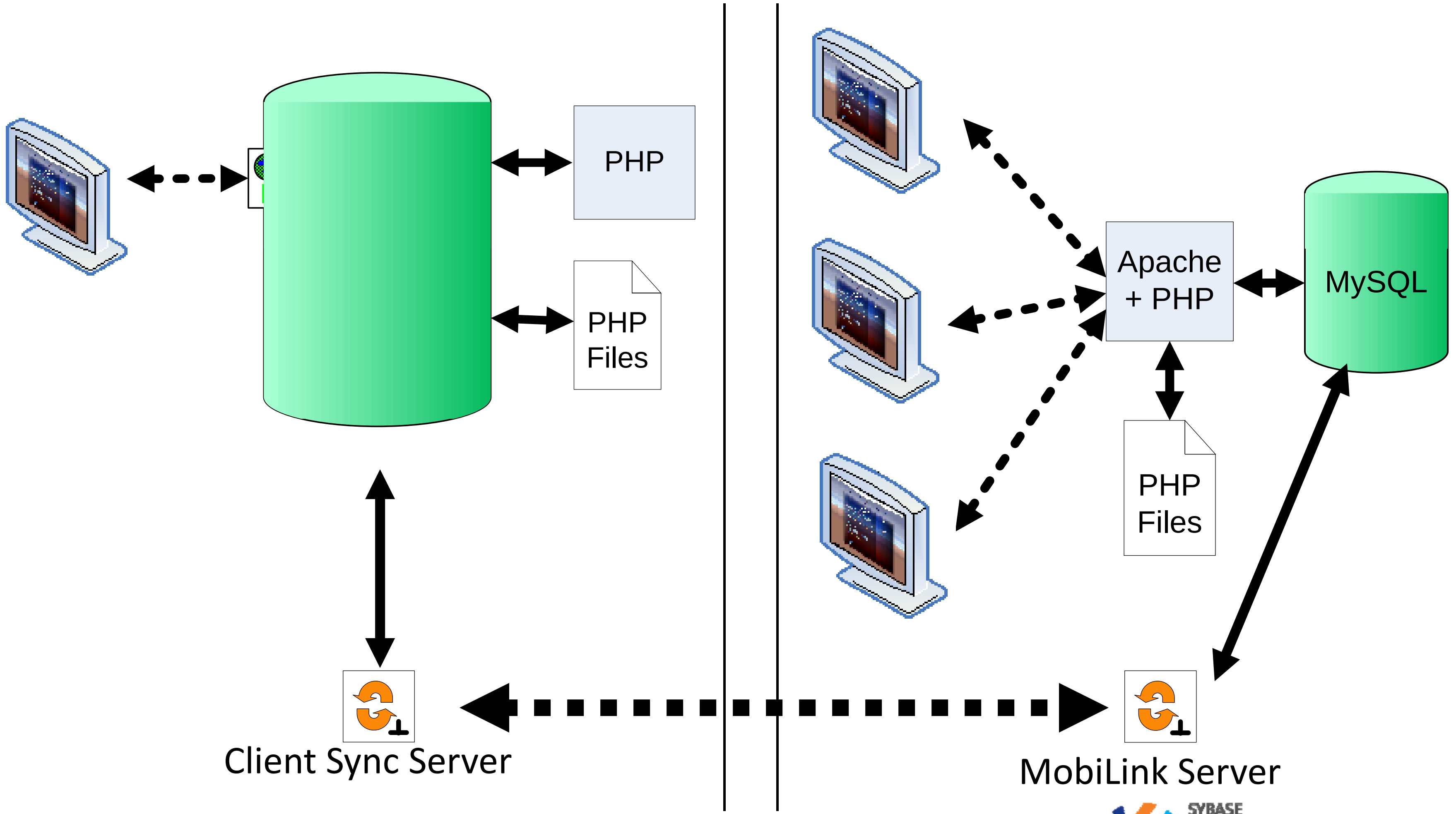




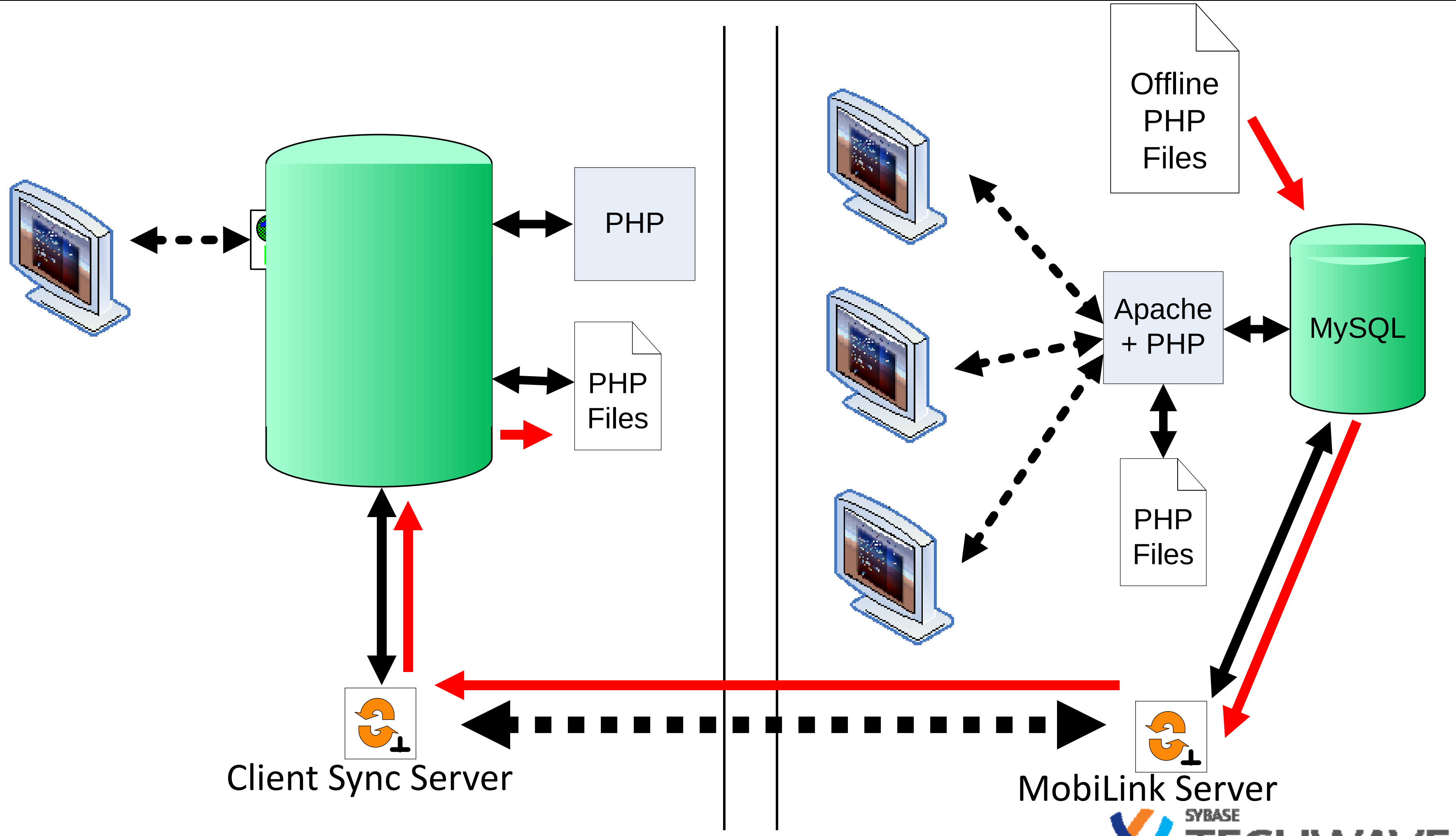
Client Sync (dbmlsync)

- Uploads all changes to database since last sync
- All synchronization happens in a SINGLE transaction
    - Any failures will cause a full rollback
- Sends both current value, and original synced value of rows
- Can be run on demanded, or scheduled

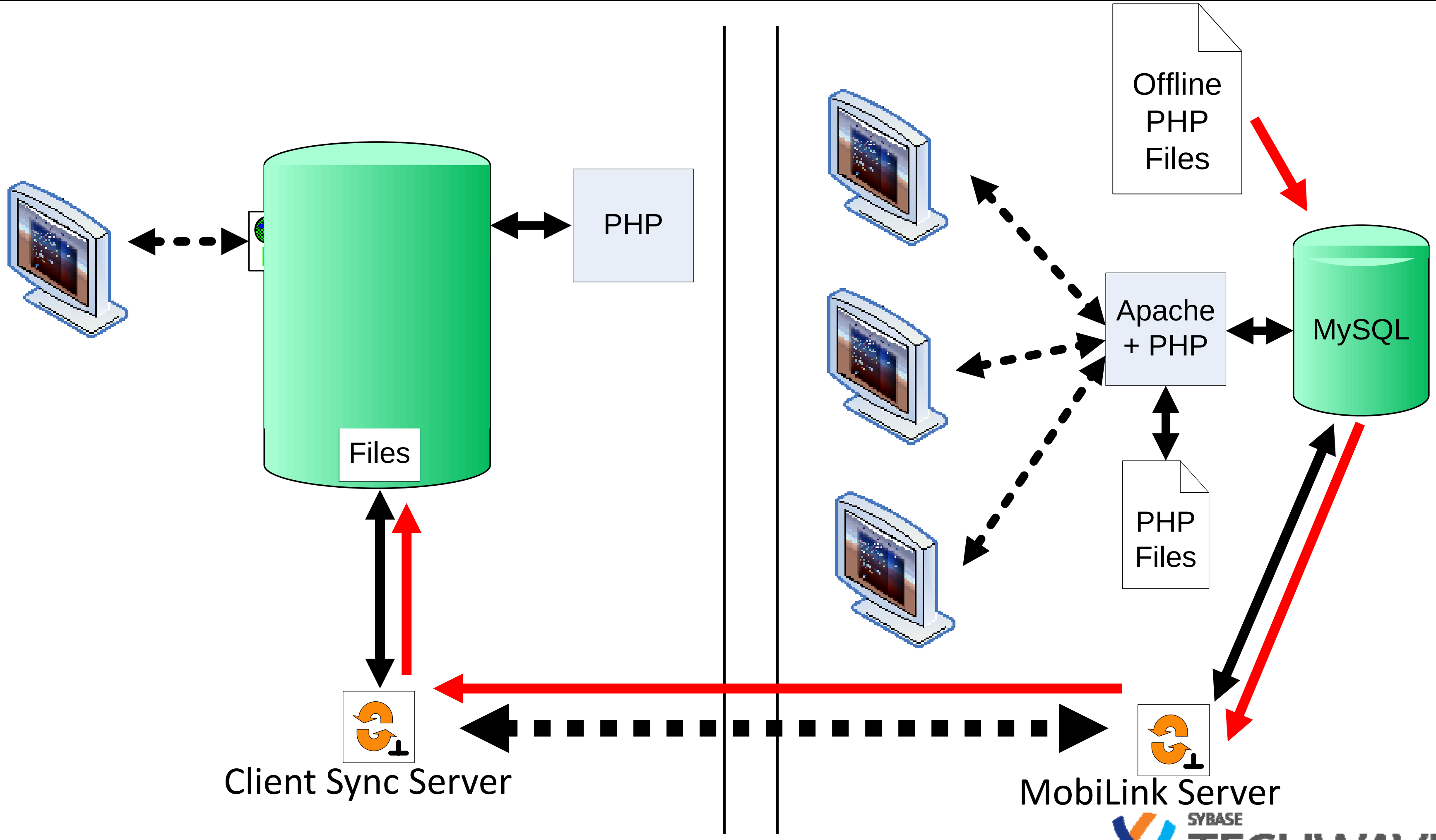
Sync!!


PHP
PHP Files
Client Sync Server
Apache + PHP
MySQL
PHP Files
MobiLink Server

Syncing Files



Storing Files in the Database



Conflicts

- What if two people change the same blog post?
  - Posts are stored a single row in the database
  - Need to ‘merge’ the rows
- Solution:
  - Use `diff3` utility as our custom business logic
  - `diff3` requires:
    - Original
    - My changed copy
    - Their changed copy

Other Languages

- Although this example uses PHP, can be used for other languages as well
  - Java
  - .NET
  - Perl
  - Python
  - Ruby
  - etc....

How can we solve the problem?

1. Take the stack offline

2. **Rewrite the application as a client-side application**

Some Definitions

- Server-side scripting languages run at the server
  - Ex. PHP, Perl, Ruby
- Client-side scripting languages run in the browser
  - JavaScript
  - ActionScript (via Flash Player)

Some More Definitions

- HTML5
  - Specification for the next version of the HTML standard
  - Currently in draft by the W3C
    - 2012 - Candidate Recommendation
    - 2020 - Final Working Draft
    - 2022 - Proposed Recommendation
  - Some browsers (Chrome, Firefox) already implement a few of the features
- Gears
  - Browser plugin/add-on
  - Open source project originally created by Google
  - Goal is to implement a subset of the HTML5

The Gears' Modules

- Implements three main features:
  - LocalServer
    - Caches resources (HTML, JavaScript, CSS, JPG, etc) and serves them when no connection is available
  - WorkerPools
    - Creates psudo-threads for JavaScript
  - Database
    - Stores local data
    - Gears currently uses SQLite
- Everything is there to make an application, but how do you synchronize your data?

Disclaimer

Statements concerning Sybase iAnywhere's new products are forward-looking statements that involve a number of uncertainties and risks and cannot be guaranteed. Factors that could ultimately affect such statements are detailed from time to time in Sybase's Securities and Exchange Commission filings, including but not limited to its annual report on Form 10-K and its quarterly reports on Form 10-Q (copies of which can be viewed on the Company's website).

All of the information in this presentation after this slide is forward-looking as defined above. As such, there is uncertainty associated with if or when any of these features will be added to the product.

SYBASE TECHWAVE SYMPOSIUM 2009

UltraLiteWeb

- A alternate build of Gears with the addition of an UltraLite module
- Allows Gears application to leverage the full power of MobiLink synchronization

```
// ----------------------------------------------------------------------------
// ** Creating and Opening an UltraLite Database **
// ----------------------------------------------------------------------------
// Create an instance of the 'beta.uldatabase' object
// using the Gears factory
var db = google.gears.factory.create('beta.uldatabase');

// UltraLite databases can be created/opened with a variety
// of configuration options. These options are passed in through
// a dictionary object
var options = {
  case: "Respect",     // Respect case sensitivity
  page_size: "8k",     // 8K memory pages
  date_order: "MDY"    // Interpret data in order of 'Month-Day-Year'
};

// Open (or create) a database called 'database-test' with
// the supplied options
db.open('database-test', options);
```

UltraLiteWeb

```
// ----------------------------------------------------------------------------
// ** Simple Queries **
// ----------------------------------------------------------------------------
// Create a table with an auto incrementing primary key called "id",
// and a 32-character column called "name"
var sql = "CREATE TABLE \"Foods\" (" +
          "  id int primary key default autoincrement," +
          "  name char(32)" +
          ")";

// Execute the query
db.execute(sql);
```

UltraLiteWeb

```
// ---------------------------------------------------------------------------
// ** Preparing for Synchronization
// ---------------------------------------------------------------------------
// Create a synchronization profile for synchronizing with a server-side
// database. The business logic of the synchronization is defined at the
// the server. Each profile synchronizes with series of publications. A
// publication is predefined set of tables and business logic.
sql = "CREATE SYNCHRONIZATION PROFILE MySimpleProfile '" +
      "Stream=https(host=my.mobilinkhost.com;tls_type=rsa);" +
      "ScriptVersion=Food_v1;" +
      "MobiLinkUid=ml_user;" +
      "MobiLinkPwd=ml_user;" +
      "Publications=FoodPub;'";
db.execute(sql);

// Synchronization happens asynchronously. The user-defined "onsync"
// function will be called with periodic updates as the synchronization
// proceeds.
db.onsync = function(status) {
  console.log("Current Sync State: " + status.state);
}

// ---------------------------------------------------------------------------
// ** Starting a Synchronization
// ---------------------------------------------------------------------------
// Start the synchronization using the "MySimpleProfile" profile
db.sync("MySimpleProfile");
```

UltraLiteWeb

- More to come. Stay tuned....
- Watch my blog for announcements

Taking It All Offline

1. Leverage your current codebase and take the stack offline
   - Allows code reuse on the client application
   - Requires SQL Anywhere installation on the client machine
   - May require a server-side scripting language installation
2. Rewrite the application as a client-side application
   - Requires a browser add-on
   - Application must be totally written in client-side languages (JavaScript, CSS, HTML)

Where to get more information

- SQL Anywhere PHP Developer Center
  - http://www.sybase.com/developer/library/sql-anywhere-techcorner/php
- SQL Anywhere Web Forum
  - http://groups.google.com/group/sql-anywhere-web-development/
- SQL Anywhere PHP Module page
  - http://www.sybase.com/detail?id=1019698
- My Blog – “Peering Behind the Browser”
  - http://iablog.sybase.com/efarrar