



Coherent Intranet developments with CakePHP

Yann Le Blevec
CakeFest @ Orlando ,FL
6 -8 Feb. 2008

Agenda

- Context and motivations
- Using a framework for improving our web developments
- Extending CakePHP
 - GUI enhancements
 - Generic software components
 - Connections with others systems
- Creating a coherent whole from several independent web applications
- Conclusion

The image features a black background with several abstract geometric elements. A thick vertical orange line runs along the left edge. A thick horizontal orange line runs across the top. A smaller horizontal brown rectangle is positioned above the main text. A square brown rectangle is located to the left of the main text. A thick vertical orange line runs along the right edge. A horizontal orange line crosses the right side of the page, intersecting the vertical orange line. The text 'Context and motivations' is centered in white.

Context and motivations

Context

- About me
 - PhD student in computer science
- My company
 - Subsidiary of Alcatel Lucent, [Annecy, France](#)
 - Computer science not core business
- Reduced IT team
 - 10 people with 1 full time web developer (not me)
- Web development
 - Mainly Intranet applications (4 already in CakePHP)
 - Windows, Apache, Mysql, PHP infrastructure

Development constraints

- High flexibility so as to adapt to business needs
 - Applications serve business
- Not much performance issue
 - Few application users (< 1000 worldwide)
 - Small sized applications
- Maintain simultaneously several applications

Needs

- After several years of PHP development
 - Lack of reusability
 - High complexity / heterogeneity
 - Difficulty for maintenance
- Highlight strong needs
 - Methodology – another (critical) issue
 - Framework

The background is black with several decorative elements: a vertical orange line on the left, a horizontal yellow line at the top, a brown square overlapping the word 'framework', another brown square overlapping the word 'web', and a vertical yellow line on the right with a horizontal orange line crossing it.

Using a framework
for improving
our web developments

Available solutions

- Many solutions have been released recently
- Moving to another language
 - Ruby On Rails
 - And also J2EE, Python Django, Smalltalk Seaside...
- Staying with PHP
 - CakePHP
 - And also Symphony, Zend framework...

Choice of CakePHP

- Ruby on Rails (RoR)
 - **Ruby** - Pretty syntax
 - **Rails** - Simple code (DRY), design patterns (MVC), best practices (convention over configuration, FS structure)
 - Blocking issues for us
 - New language to be learned by web developers
 - Missing administration capabilities (new Web AS)
- CakePHP
 - Most of Rails enhancements with less migration problems
 - CakePHP 1.1 over 1.2 for stability reason

Required application components

- CakePHP is a generic framework
 - **Must** bring as many common functionalities as possible
 - **Can** provide optional pluggable elements (bakery)
 - **Can't** solve all issues (client specific implementations)
 - User management, layouts ...
- CakePHP is not the optimal start point for our Intranet developments
- Expand CakePHP so as to create a custom template
 - Consider the specificities of our environment
 - Evaluate our needs



Extending CakePHP

Part 1. *GUI enhancements*

GUI enhancements

- A custom layout for all applications
 - Follows company's communication recommendations
 - Generated directly by a modified '*bake*'
- Helper for gathering smart functionalities
- GUI components
 - Contextual help
 - Lite date picker (<http://calendar.swazz.org/>)
 - DHTML date picker for human beings
 - JComplete
 - Select one or more values among a list of possibilities

Jcomplete - Description

- Inspired by 'Tag completion' (*delicious*) and Auto-complete functionality (bakery)
- Mainly based on two Javascript scripts
 - One generated on the fly for the data
 - One for rendering the component (some of delicious code)
- Simple to use
 - *View* – Use a helper to add completion to a text field
 - A maximum number of inputs may be specified
 - (Plugin) *Controller* – Code data retrieval for completion
 - Bring security and flexibility to the component

JComplete – Code sample

- View

```
$customHelper->init_multi_completion( array (  
    // ADD COMPLETION ON AuthorDocument TEXT FIELD WITH DATA TYPE allowedAuthor  
    'AuthorDocument' => array ( 'sugDiv' => 'suggestDiv_1', 'type' => 'allowedAuthor' ) );
```

- Plugin > Controller

```
class TypeLoaderController extends JcompleteAppController {  
function get() {  
    ...  
    switch ($type) {  
        case 'allowedAuthor' :  
            // CODE FOR BUILDING THE LIST OF AUTHORIZED AUTHORS  
            $res = $this->User->generateList('User.authorizedAuthor = 1', null, null,  
                '{n}.User.cn', '{n}.User.displayname');
```





Extending CakePHP

Part 2. *Generic
software components*

Generic software components

- Recurring sections
 - Welcome page
 - Administration of the web application
 - Help/Support – direct bug report to TRAC
- Database initialization scripts
- User / Group management
 - MVC objects for '*Users*' and '*Groups*'
- Permission management
 - MVC objects for '*ControlObjects*'
 - ***Aclite*** – Layer built upon ACLs for a simpler usage

Aclite

- Disappointed with tested solutions
 - ACLs in CakePHP 1.1
 - Incomplete for a direct integration
 - *phpgacl*
 - Component oversized (too much code and tables)
- Development of **Aclite**
 - Based on ACLs
 - GUI for permission management
 - Map requester with controller objects
 - Component for checking automatically authorizations

Aclite – Design time

- Controllers declare static permission
 - List of required control objects
 - Fine grained actions (CRUD) are available
 - Global or/and local declaration
 - Defined in 'AppController' or/and in each controller
- Users may belongs to 0..n groups
 - '*Anonymous*' group is reserved to unidentified users
 - '*Member*' group is reserved to identified users
- Groups have predefined privileges on control objects
 - 'aros_acos' table

Aclite – Design time



```
var $authLocal = array(  
    'Users' => array('authorizations'),  
    'except' => array (  
        'login' => array('public'),  
        'logout' => array('public')  
    )  
);
```

Aclite – Run time

- Authorizations checks
 - Verify if the user belongs to a group with sufficient privileges on the requested control object
 - *If correct*, the request is forwarded to the action
 - *Else*, the user is redirected to a 'Permission denied page'
- Dynamic permissions (advanced usage)
 - Aclite provides a callback function *beforeAuthorize()*
 - *Example*: Only document owners can edit...
 - Possibility to create dynamic memberships
 - Dynamic groups are stored in sessions



Extending CakePHP

Part 3. Connections with
other systems

Connections with other systems

- Intranet applications exist within a system landscape
- Keyword is *integration*
 - LDAP (Active Directory and Openldap), ERP system (SAP), third party applications (middleware)
- Implemented functionalities
 - User authentication based on Web Services
 - Suppling/consuming data to/from SAP
 - User connected to LDAP (no '*users*' table)

Web service technology

- No CakePHP applications act as a service provider yet
 - Required in a near future
 - Depend on REST / SOAP support in CakePHP 1.2
- Web service implementations
 - SOAP document/literal
 - Programming languages involved
 - PHP 5 SOAP extension but also Java J2EE, delphi
 - Architecture organized around a central message HUB
- Example of the authentication service
 - Validates credentials against several directories

LdapSource – What for?

- Why is it so important to retrieve user data directly from LDAP?
 - LDAP is the central repository for all employee
 - Contain up-to-date information
 - Dedicated service
 - All employees are potential application users
 - No need to duplicate user information in a database table and rely on a synchronization script
- LdapSource allows us to work on fresh user data
 - Use LDAP models that look like all other cake models
 - LDAP users have an ID used as a key in tables

LdapSource – A datasource

- ***Datasource*** in CakePHP
 - Abstraction layer between models and data
- Wide database support through the DBO datasource
 - MySQL, sqlite, AdoDB, ODBC , PEAR,...
- What about LDAP?
 - Not only a matter of authentication
 - No conceptual differences with data coming from a database
 - Contain information about users, groups, sections....
 - LDAP datasource in CakePHP 2.0?
- LdapSource: our implementation of a LDAP datasource

LdapSource – Details

- Setting up your CakePHP application
 - An additional database must be configured in '*database.php*'
 - Several variables must be defined in LDAP models
 - ***useDbConfig***, ***primaryKey*** (ID), ***useTable*** (branches w/o base)
- Supported features
 - LDAP Models Provide all 'findAll' based operations
 - Most query parameters (fields, limit, order) are processed
 - Use any associations on or with a LDAP model
 - Ex 1. A user belongs to group, a group has many User
 - Ex 2. A document has and belongs to many author (type user)

LdapSource – Details

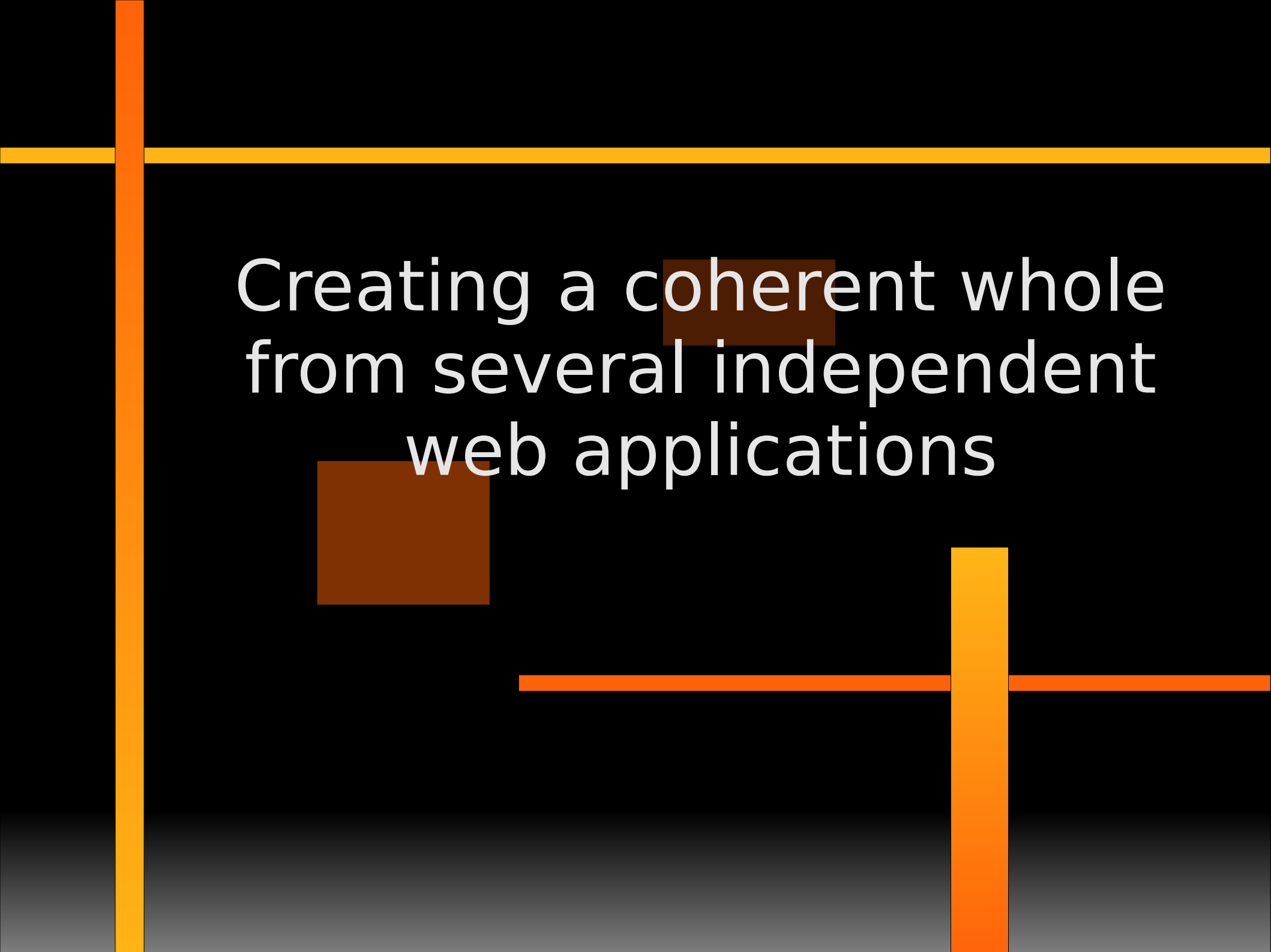
```
var $ldap = array (  
    'datasource' => 'ldap',  
    'host' => 'hostname',  
    'port' => 389,  
    'database' => 'dc=myCompany,dc=fr',  
    'login' => 'cn=admin,dc=myCompany,dc=fr',  
    'password' => 'admin',  
    'version' => 3,  
    'cache_expires' => '+24 hours'  
);
```

LdapSource – Cache

- LdapSource also provides caching functionalities
 - Performance optimization
 - Ex. Many pages that require the same list of users
 - Implemented with CakePHP Cache object
 - Cached LDAP models must define ***\$cachedFields***
 - Specify which LDAP fields need to be cached locally
- Conditions for a search query to hit the cache
 - *Explicitly* : Fields parameter equals to '***__\$cache__\$***'
 - *Implicitly* : No condition and no limit have been defined

LdapSource – Missing features

- LdapSource is **not yet finished**
- Remaining tasks
 - Reorganize source code
 - Enhance query syntax
 - Implement missing functionalities
 - Create, update, delete and query
- Contribute, comment and get more details with the bakery website
 - <http://bakery.cakephp.org/articles/view/ldap-datasource-for-cakephp>

The image features a black background with several abstract geometric elements. A thick vertical orange line runs along the left edge. A thick horizontal yellow line runs across the top. A smaller horizontal orange line is positioned below the yellow one. A vertical yellow line runs down the right side. A horizontal orange line crosses the vertical yellow line. A brown square is located to the left of the text, and another brown square is positioned behind the word 'coherent'.

Creating a coherent whole
from several independent
web applications

Great applications... not enough!

- As a *developer*
 - Our custom template brings many improvements
- As an *web administrator*
 - Still support many web applications
 - No significant improvement
- Administrator need
 - Gather centrally some administrative features
 - Bring reusability at administration level
- Creation of a coherent whole

Common administrative features

- ***Fredistrano*** – web deployment tool
 - Send source code from a subversion repository to production servers
- ***Message broadcasting***
 - Send live notifications to business application users
- ***Log viewer (currently in development)***
 - Monitor centrally all applications logs
 - Highlight data according to their business significance
- ***Application probing (currently in development)***
 - Detect unavailability or bad performance

Deployment with Fredistrano

- Deployment tool for PHP web applications
- Core features
 - Export source code from subversion to production server
 - Synchronizes the exported revision with a target directory
 - Remove the burden of several manual tasks
 - Rename configuration files, modify file permissions...
- Technical details
 - CakePHP application
 - Run both on Windows/Linux environment
 - Cygwin required with Windows

Message broadcasting (1/2)

- *Use case*
 - A new version of an application is about to be deployed
 - Service need to be temporary suspended
 - Active users should be notified (phone? mail? message!)
- *Aim*
 - Render message directly inside target applications
- *Solution*
 - Application for managing messages
 - Periodical AJAX calls from the browser of business application users to the broadcast application (*pull*)

Message broadcasting (2/2)

- Important message properties
 - **Active** – Tell if the message should be sent
 - **Validity** – Date after which a message is mark as deprecated
 - **Type** – Several types are available indicating how applications should render the message (the more critical, the more intrusive)
 - **Target application** – Associated application names (all is possible)



Conclusion

Conclusion – *1 year of CakePHP*

- ***Overall feeling is great***
- ***Major benefits***
 - *Cleaner code* – easier to develop, simpler to maintain
 - *Focus on business process* – less time spent on technical issues (ex. MySQL queries)
- ***Some problems***
 - More time spent on code optimization (ex. bind/unbind)
 - Difficulties to report template modifications to previous applications
 - As many template instances as business applications
 - No package management (Slice?)

Next ?

- Move our template to CakePHP 1.2
 - Benefit from major enhancements
- Continue to centralize as many shared features as possible
 - Global permissions management
- Try to contribute to the community by releasing some of our source code

Questions?

- Thank you for your attention
- Slides of the talk available at
 - http://cakephp.fbollon.net/cakefest_2008/
- Contact
 - email_ylb-php@yahoo.fr