Technology Overview

FARCRY 2.1 WHITEPAPER

Prepared by Geoff Bowers, Systems Architect
Daemon Pty Limited

14 May 2004

Version 1.0

1 Table of Contents

2 Introduction

2.1 TARGET AUDIENCE

The Technology Overview is best seen as a helicopter view of the entire FarCry application framework. It's targeted at developers and evaluators trying to get a grip on the concepts, approaches and vocabulary used within FarCry CMS. It should not be viewed as a definite explanation of all features and services rather this document aims to deliver an overall vision of FarCry capabilities and strengths, and is an ideal primer for developers looking to get involved with the code base.

2.2 FARCRY CMS PLATFORM

2.2.1 Enterprise Class Solution

FarCry CMS is an enterprise class, open source code base. It is the largest ColdFusion, open source code base in the world. FarCry's success as a commercial content management system has been eclipsed by the rapid growth and uptake having moved to open source in April 2003. A detailed outline of FarCry features (FarCry FAQ) from a business requirements perspective can be provided on request.

2.2.2 Highly Extensible Framework

FarCry CMS is a flexible and highly extensible application framework for building content rich, web applications. The FarCry code base has been specifically architected to allow for extensive modification whilst still retaining the ability to update the core library of code and participate in ongoing feature development.

2.2.3 FarCry Community

FarCry CMS is more than just an open code base. Daemon has invested significant resources to provide services and infrastructure to allow the FarCry community to flourish. The health of the surrounding developer community is integral to our overall business strategy and as such has been a focus from day one. The FarCry community is a primary driving force behind feature development and support and solutions broad uptake ensures an innovative future moving forward.

2.2.4 Open Source & Technology Executive Overviews

For a non-technical, perhaps more executive oriented overview of working with FarCry you may find the following presentations of interest:

FarCry: Planning for Open Source Success
Minimise the risks of implementing open source solutions in your enterprise with a commercially supported FarCry community:
http://daemoninternet.mmreseller.breezecentral.com/p24761619/

FarCry CMS Technology Overview
FarCry is a proven enterprise solution released to open source. This gives the code base a maturity and discipline often missing in community development efforts.
http://daemoninternet.mmreseller.breezecentral.com/p77866929/

3 System Overview

3.1 PROGRAMMING ENVIRONMENT

FarCry is built on top of the award winning Macromedia ColdFusion application server. ColdFusion is a Sun Certified java application. ColdFusion script and mark-up compiles to java byte code and runs on the majority of J2EE application servers. Although the underlying Java environment can be leveraged, FarCry developers only need to understand the CFML language in order to develop within the application framework.

3.2 DYNAMIC AUTHORING AND PUBLISHING

FarCry CMS follows a dynamic authoring and publishing model. Content is composited on the production server and delivered to the visitor "on the fly". FarCry does not publish static HTML files. Although FarCry does have a series of sophisticated caching services that can be used to reduce load on the application server as required.

Content authoring, staging and delivery can all be served from a single server instance. Alternatively these functions can be distributed across multiple servers depending on the deployment. FarCry features a virtual-staging mode that allows contributors to preview draft content in the context of the live, published website.

The authoring environment is feature rich and a more comprehensive discussion of authoring and publishing can be found in the FarCry Feature FAQ.

3.3 CONTENT OBJECT API (COAPI)

The FarCry CMS Content Object API (COAPI) forms the foundation for all content built and managed on FarCry CMS. The COAPI is a database abstraction layer that uses an object-based programming model for managing content. The site object model is a blueprint for the underlying data schema of any FarCry application.

The COAPI provides a content storage model that enables developers to transparently store all forms of content in a structured object storage system. The FarCry Content Object database is built on top of a standard relational-database model, enabling customers to use almost any standard relational database as the actual storage infrastructure. Support as of v2.1 includes mySQL, MS SQL Server, Oracle with PostgreSQL slated the for v2.2 release.

Storing content in a core repository meets the key design goal of a clean separation of data or content from presentation. This allows developers to easily repurpose content based on a given user's browser, or for delivery using other formats, such as XHTML, XML, RTF, e-mail, etc.

Using the COAPI shields developers from having to program or manipulate the database schema, instead they use an elegant ColdFusion component-based API to interact with the Content Object database. This API facilitates the deployment of new content types and the modification of existing types. In addition it manages the evolution of the underlying relational database schema, including the modification of data types.

Because the COAPI is extensible to support any form of back-end system, developers can simultaneously access content stored in either the Content Object database or in their own external databases. The FarCry Content Object database natively supports storing most popular content types, including files, images, Flash animations. Binary media such as files, images, and audio/video are stored in file-system based media directories for fast retrieval and delivery. Developers can also extend the ContentObject database to include content and data stored in any external databases.

3.3.1 Content Types

A site object model is comprised of a collection of Content Types. Each object type contains a set of properties and methods. Properties are the elements of data associated with an object type. For instance, a news object type might have title, teaser, body and publishdate properties. Methods are the activities that surround the object, such as edit and display. The code that manages the activities is built using a ColdFusion component. The properties of information or data can be stored and managed by FarCry, using the Content Object database, or in any external RDBMS accessed via SQL or object middleware.

Core types are those content types that ship as part of the FarCry core libraries. They include Navigation, HTML, News, Events, Files, Images and many others. These by and large make up the default installation and administration interfaces within FarCry CMS. A "vanilla" FarCry deployment is one that only uses the FarCry core types and has no customisations. Custom types are content types that are specific to the FarCry application in question. This term can also refer to core content types that have been extended and modified.

Content types are based on ColdFusion components. These specialised components extend (or inherit) an abstract class (types.cfc) within the FarCry core library that provides the component with system properties and methods needed to operate within the FarCry framework. This allows custom objects to immediately leverage the plethora of services within the FarCry framework. Obviously specific properties and methods can be overridden as required.

3.3.2 Content Objects

Content object is a term used to represent a specific instance of a content type. For example a News content type would define the relevant properties and methods available – like a boiler template. The content object would represent a specific instance of news content. Although all content objects of a particular type will have different data associated with them, they will nevertheless share a common set of property fields and methods.

3.4 TREE MODEL

Positional information within the FarCry CMS environment is modelled in SQL using a variation of the nested tree model approach (championed by Joe Celko, http://www.celko.com/). This provides a very fast and powerful mechanism for managing tree related information. For example, the site information hierarchy, the relationship of one fixed page to another and the categorisation service, the relationship of one keyword to another, are managed within the FarCry tree model.

The tree modelling approach allows for all kinds of sophisticated content management including the ability to easily rearrange positional information and rapidly determine essential metadata such as breadcrumbs, ancestry, descendents and so on. Developers don't need to understand the tree model to leverage its power – everything is exposed through a simple component API. Common tasks like generating navigation and positional information for the presentation layer are simplified further still via a library of custom tags.

3.5 SERVICES

FarCry, as you might expect, contains a library of services for handling content management functions. These include workflow, versioning, Verity free text search integration and so on. The FarCry administration area is effectively a sample application that leverages these services to provide the visible out of the box FarCry solution for the default content types. These services can also be leveraged to support custom content types, publishing rules and other customisations. Effectively anything that is possible in the default administration area should be accessible as a service through the component API.

Components are generally self documenting, and it's worth using the ColdFusion component browser to review the functions available by default in the application framework. Many integration examples are available for review from the FarCry developer community.

3.5.1 Versioning and archiving

By default only HTML objects are versioned and archived, although these services can be leveraged for any content type within the system.

3.5.2 Verity free text search services

FarCry leverages the built-in Verity K2 engine shipped on the ColdFusion application server to provide free-text searching across all content types. The individual content type properties you require searching for can be configured at will through the web based administration. FarCry also provides options for searching external file libraries.

3.5.3 Scheduled tasks

FarCry has its own task management subsystem that uses the underlying ColdFusion cron service. This is typically used for scheduling maintenance events, periodic reporting and the like.

3.5.4 Categorisation

There is a central categorisation service that can be applied to any content object. The categorisation service is exposed by default to many of the core content types. Categorisation is based on a keyword tree that can be edited by appropriately privileged users. Content can be selected for by specific keyword allocation or by keyword tree branch.

3.5.5 Export

FarCry ships with an export service that allows the content instances of any content type registered in the system to be exported according to a developer defined template. By default we incorporate a basic XML template and a more specialised RSS template for content syndication. Developers can of course extend this service by providing additional templates as required.

3.5.6 Reporting

FarCry has an observation architecture that records the detail of every object access. Detailed web statistics reports are available for all kinds of activity throughout the website. FarCry ships with a standard set of reports that can be supplemented as required by processing the captured data.

FarCry applications generate a standard web log (from the webserver itself) for third-party logging tools to process.

3.6 SECURITY MODEL

FarCry incorporates a very sophisticated role based security model. FarCry authenticates against a user directory expecting a true/false on the login credentials and group membership for the user. User groups are then mapped to internal policies or roles, thereby determining the privileges of the user in the system.

3.6.1 User directories

FarCry supports multiple user directories including ActiveDirectory, NTDomain, LDAP and JDBC/ODBC directories. Significantly FarCry can manage users in multiple and mixed directories at once in a single application. Furthermore, user directories can be shared across multiple FarCry application instances.

3.6.2 Policy Groups or Roles

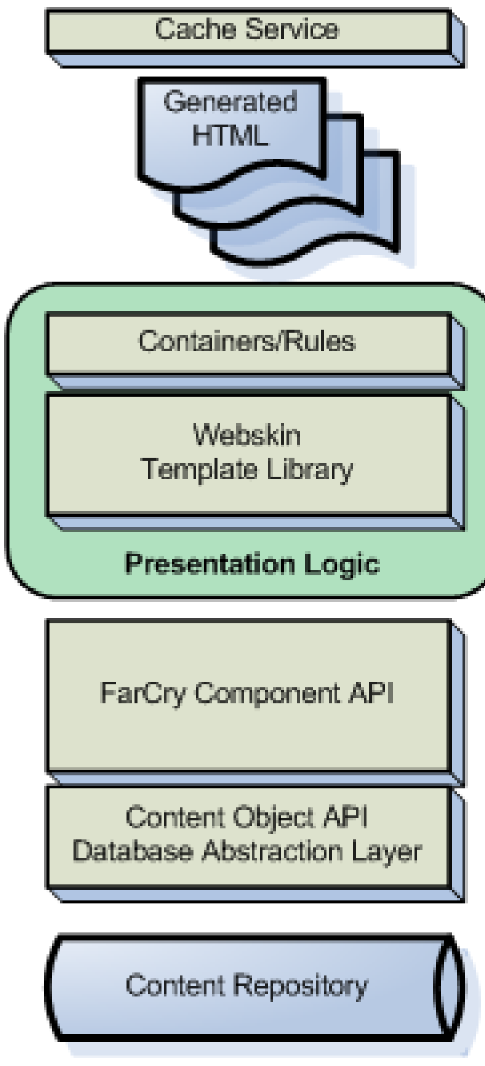
FarCry policy groups are effectively a collection of permissions within the system. FarCry has a library of permissions covering all aspects of the framework. Policy groups are mapped to specific user groups within the associated user directories configured for the application.

Policy groups are completely customisable. FarCry ships with five default policy groups. These can be modified, substituted or added to as required. “Anonymous” is a special role identified to the system as effectively the absence of any group information – its very handy to be able to apply permissions for users who have not been authenticated.

Developers can readily add additional permissions to cater for customisations they may need to make.

In addition, developers can use permissions to provide visibility on tabs and menu items within the web based administration as well as access to specific functions or actions.

3.7 PRESENTATION LOGIC

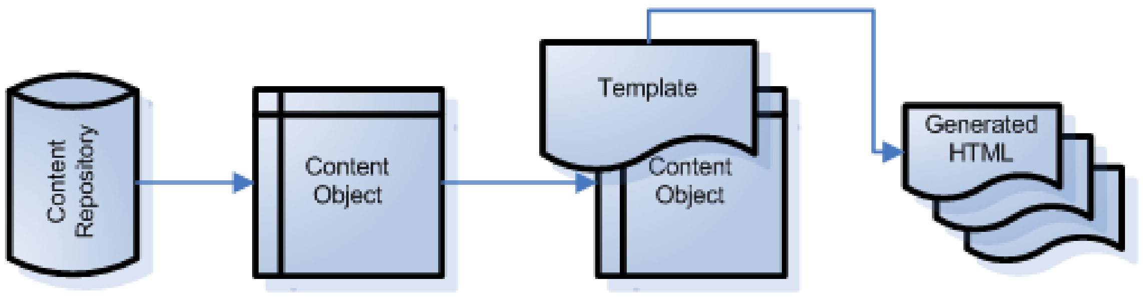


FarCry isolates the presentation layer into a special area of the code called the "webskin". We've done this to provide the most simplistic offering possible for template designers to composite presentation templates within the system. As such, templates can be developed by anyone confident with HTML.

Literally any sort of presentation layer can be developed. By way of example only, FarCry installs with a set of XHTML compliant templates using Ben Bishop's Aura (http://www.leorex.com/products/aura/) template design.

Templates are ColdFusion pages. As such they can incorporate any functionality available to the FarCry component API or the underlying ColdFusion server itself. Consequently a template in the FarCry sense can do all sorts of interesting things. The template code has access to the properties of the object instance, its positional information (including an understanding of its parents, children, siblings and related links) environmental variables (including any session profile for the current user), and all the underlying FarCry services. This essentially allows for an extremely rich environment for the development of the presentation layer. For example, a page breadcrumb can be dynamically built by asking the FarCry API where in the site the currently viewed content sits.

Each content type, for example a web page or news article, can have any number of templates associated with them. For instance, a web page content type might have templates for "Full Page Display", "Teaser Display", and "Text Only Display". A single content item could be rendered in several different ways, just by changing the processing template.



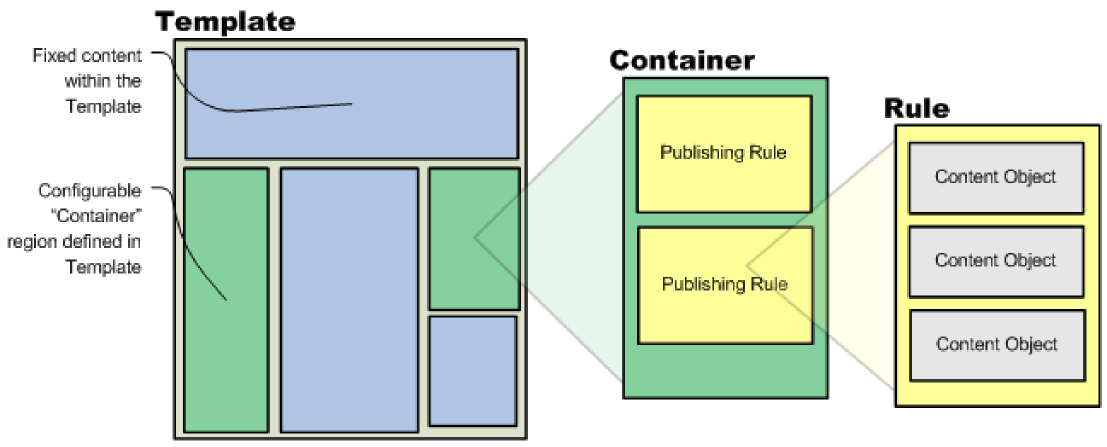
3.7.1 Template Caching

FarCry has a template caching sub-system that allows template designers to cache different regions of the template for any specified time. For example, one might cache the primary navigation for 6 hours but a random testimonial for only 5 minutes. Any combination of caching regions and periods are possible. Creating a cached region on a template is as simple as enclosing a section in a ColdFusion custom tag. The FarCry web based administration allows for the flushing of cache blocks on demand or programmatically across the entire web site.

3.7.2 CSS support

FarCry has a specific content type for managing external style sheets. This can be referenced in the template to provide a specific set of cascading style sheets for different regions of the web site. Furthermore, these style sheets can be managed independently of the template engine by non-technical users through the web based administration area.

3.8 CONTAINER MANAGEMENT



FarCry has a special sub-system for managing dynamic content within the presentation layer. Specific regions of any template can be designated as a "container" and have dynamic content scheduled into them by non-technical authors. This behaviour in other systems is sometimes called, portlets, page behaviours, pods or a variety of other names.

The concept is important to understand because it represents some of the most powerful aspects of the FarCry framework. Containers are regions of a page where some sort of programmatic behaviour can be placed. For example, a list of the latest news items, an event calendar, content syndicated via XML from another website, a randomly picked fact and so on. These programmatic behaviours in FarCry are called Publishing Rules.

3.8.1 Containers

Containers are placed in a template with a simple custom tag. FarCry takes care of the rest of the required machinery. Based on the name given to the container, they can be unique to a specific page, shared across a specific section of the website or shared globally throughout the website. They are populated by contributors selecting from a predefined set of publishing rules.

3.8.2 Publishing Rules

Publishing rules are a special content type that capture some parameters in edit mode from a contributor and then at run-time, in execution mode, dynamically display content based on some predefined behaviour. FarCry ships with a library of publishing rules for contributors to choose from. Developers can customise these rules or build their own.

For example, the news rule allows a contributor to select a type of display, categories for filtering and the number of news items. When a visitor accesses the page with the rule, the predefined behaviours goes off and grabs the very latest news items matching the specified categorisation and displays them with the right template. When even new news content is added to the system, this rule will automatically update the page without any further intervention.

For example, the child links rule will display teasers from all a pages underlying child pages. This sort of behaviour is used to populate a summary or landing page dynamically based on the underlying pages of that section of the website. Add, update or remove a page and the landing page is automatically updated.

For example, the "random fact rule" will randomly pick and display a predefined number of fact content objects. This rule would update and refresh its content every page request. This is often used as a great way to show off testimonials or other snippets of information.

3.9 CODE BASES

One of FarCry's distinct architectural advantages is the separation of core CMS services from unique project by project customisations. We work exceptional hard to make the core libraries configurable by extension and not be replacement. You should not have to make modifications to the core library as part of normal development. This significantly minimises the impact of core changes on existing deployments. Furthermore, it ensures that development teams can continue to benefit from the fruits of ongoing open source development in the core code whilst providing uniquely customised deployments.

3.9.1 FarCry Core

farcry_core is the primary code base. It holds all the content management services and methods. It effectively provides the application framework and the presentation layer for the FarCry Administration area. It has been specifically isolated as the core, managed code base -- you should never have to modify this code base except to patch bugs or extend core functionality. All project specific customisations should be done elsewhere!

The core library contains all the generic services that make up a "vanilla" FarCry deployment. It is worth noting that the sample application that comes as part of the installation only represents a small sub-set of functionality available overall. The Component API is well documented, and you can review this online at: http://farcry.daemon.com.au/cfcdoc/

3.9.2 FarCry Project

Every FarCry instance has its own unique project directory. This directory holds the site specific configuration files, the webskin (presentation layer) and anything else that makes a project unique.

On installation, new applications are populated with templates from farcry_aura. You start with this very basic application structure and then modify it to your hearts content. This repository contains basic templates such that your project is operational on deployment. This basic structure includes configuration files, directory structures for customisation and a presentation layer for the core CMS elements.

3.9.3 Farcry Aura

This code base is used for installation only. The installation routine generates a new application using your choices and the base templates stored in farcry_aura. Aura is not a standalone application, just a template for building one.

3.9.4 FourQ

fourq is the content object API (COAPI) used by FarCry as a database abstraction layer. It is this abstraction layer that makes the deployment and evolution of content types within the framework so flexible. fourq was released independently to open source some time ago (http://fourq.org/) and can be used in complete isolation to power other applications not based on FarCry. In any event, its an interesting use of CFC concepts and well worth a look -- even if its just to spark your own database abstraction ideas. farcry_core provides a web-based front end to fourq actions to make life easy for FarCry developers[1].

[1] In time fourq will likely lose its independence and be moved under the farcry_core packages to simply the deployment.

3.10 EXTENSIBILITY

FarCry CMS is more than a content management solution. FarCry is an application framework that is well suited to the construction of web applications, especially those dealing with content. The framework provides a range of integration and customisation options.

3.10.1 Templates & Webskin

Developers can implement any look and feel through the FarCry presentation engine. Templates are simplified further by being self-registering, and supplemented by a custom tag library for truly painless deployment. Template construction only requires good HTML skills and a passing knowledge of ColdFusion.

3.10.2 Included Objects

The “include” content type allows developers to run any block of ColdFusion code within the template engine of FarCry. For example, the sitemap and search pages within the sample applications are both Include objects.

3.10.3 Custom Administration

The FarCry web-based administration is customisable. Developers can supplement the existing functionality with their own administration interfaces as they require. The security model, audit and other services can all be used to extend the administration area.

3.10.4 Custom Content Types

Completely new and unique content types can be built and deployed within FarCry, leveraging the same core services available to core types.

3.10.5 Extending Core Types

Core types are custom objects that extend objects in the farcry_core code base. This powerful feature allows developers to modify the behaviour of farcry_core code without any need to change the core libraries.

3.10.6 Custom Publishing Rules

Developers can build and extend publishing rules to supplement the existing library of publishing rules within the system.

3.11 THIRD PARTY PLUG-INS

FarCry leverages many third-party code libraries, to provide a best of breed solution. These plug-ins are optional but recommended. FarCry provides out-of-the-box integration for following solutions and is adding more as the community requires. These libraries are isolated from the primary code base as they may be distributed under alternative licenses and/or their source code is managed by a separate project. The FarCry milestone builds released by Daemon provide commercial support for the integration specific versions of these code libraries.

3.11.1 Friendly URLs

The FriendlyURL servlet is used to generate beautiful user and search engine friendly URLs for any FarCry application. FriendlyURL now ships as part of the standard FarCry distribution but is turned off by default. FriendlyURL is very easy to configure and is highly recommended.

FriendlyURL is developed and maintained by Spike Milligan (http://www.spike.org.uk/).

Rich Text Editor Support

We’ve specifically designed the code base so as not to enforce any particular WYSIWYG editing model on the user. This allows you to choose and integrate your preferred rich text editor. FarCry supports a variety of rich text editors, including Site Objects, RealObjects, Ektron and others. Editor integration includes the web based administration for configuring editor buttons and functions as desired. The list of supported editors is updated regularly and is largely representative of the existing community’s desires so check the

FarCry community website. If your editor is not represented, it's easy enough to add it to the supported list.

3.11.2 qForms Client Side Validation

FarCry uses the qForms javascript library for client side validation in forms and wizards. Developers are not required to use this library for their own customisations (but heh its great so take a look).

qForms JS library is developed and maintained by Dan Switzer (http://www.pengoworks.com/).

3.11.3 geoLocator

geoLocator is a Java/ColdFusion library used to lookup country code and language from a visitor's IP address. It uses a local copy of the WHOIS database to perform fast, accurate lookups of country codes. geoLocator is useful for log analysis, internationalization, geolocation, and more.

geoLocator is maintained by Paul Hastings (http://sourceforge.net/projects/javainetlocator/).

4 FarCry System Requirements (v2.1)

FarCry CMS requires the Macromedia ColdFusion MX 6.1 application server (Standard or Enterprise): http://www.macromedia.com/software/coldfusion

4.1.1 Supported Databases

Microsoft SQL Server 7 and above
Oracle 8i and above
MySQL 3 and above

4.1.2 Supported Platforms:

Windows
Intel Pentium processor or higher
256 MB RAM (512 MB recommended)
400 MB hard-disk space

Microsoft Windows 98*, ME*, NT 4 SP6A , 2000 SP3, XP, or 2003

* Windows 98 and ME supported for development only

Linux
Intel Pentium processor or higher
256 MB RAM (512 MB recommended)
400 MB hard-disk space

Red Hat Linux 7.2, 7.3, 8.0, 9, or AS 2.1
SuSE Linux 7.2, 7.3, or 8.x
TurboLinux 8 Server (Japanese Only)

(Other flavours possible although they are not commercially supported)

UNIX
SPARC, PA-RISC 1.1 or 2.0, or POWER/3 processor
256 MB RAM (512 MB recommended)
400 MB hard-disk space

Sun Solaris 7, 8, or 9
HP-UX 11i
IBM AIX 5L 4.3.3, 5.1, or 5.2
Apple Mac OS X*

* Verity Free text searching not supported on OSX

4.1.3 Supported J2EE Application Servers

- Macromedia JRun 4
- IBM WebSphere Application (Server 4 and 5)
- BEA WebLogic Server 6, 7, and 8.1
- Sun ONE Application Server 7