BEDITA, A SEMANTIC CONTENT MANAGEMENT FRAMEWORK, MADE IN ITALY

Rosanelli, Stefano, ChannelWeb Srl, via Rismondo 2, Bologna, s.rosanelli@channelweb.it

Abstract

BEdita *(pronunciato 'bi-é-dita') è un framework per creare applicazioni Web pensato per il Web2.0, che ambisce a diventare uno strumento di riferimento utile per il prossimo Web3.0 e per il futuro Web Semantico. Il software è stato sviluppato da due aziende italiane ChannelWeb S.r.l. e Chialab S.r.l., entrambe di Bologna, utilizzando uno stack di componenti basate unicamente su software libero.*

BEdita *è disponibile con licenza Open Source/Free Software* ***Affero GPL3:*** *nell'articolo si parlerà della licenza Affero GPL3, di come questa sia stata creata proprio nell'ambito dei software per servizi via rete (come nel nostro caso) per risolvere l'annoso problema dell'ASP loophole: in pratica cambia il concetto di "distribuzione" del software rispetto alla classica GPL. Saranno analizzate le motivazioni per cui è stata scelta questa particolare licenza per il progetto. Sarà inoltre evidenziato come questa scelta potrebbe incoraggiare la condivisione di progetti software e obiettivi commerciali nel mondo della piccola e media impresa operante nel campo dell'ICT, soprattutto nel contesto italiano.*

BEdita è anche disponibile in una versione con licenza proprietaria che contiene (attualmente) alcuni moduli verticali aggiuntivi (per editoria ed e-commerce). Il modello di business adottato è quindi quello noto come dual licensing.

Nell'articolo saranno illustrate le principali caratteristiche tecniche di BEdita: disegnato e creato come framework semantico a oggetti in grado di gestire contenuti multimediali e di altra natura. Gli elementi fondamentali del framework sono due:

1. *un'applicazione web di backend per la redazione dei contenuti e delle loro proprietà e relazioni semantiche, con un'interfaccia utente innovativa per l'ergonomia tramite l'associazione cromatica fra tipologie di oggetti, l'uso estensivo del drag'n'drop e altre tecniche AJAX;*
2. *un API di frontend, servizi e specifiche per creare applicazioni frontend.*

Si tenterà di spiegare perchè parliamo di Framework e non di CMS (Content Management System), e di quali analogie e differenze ci siano fra BEdita e sistemi tipo Web Content Management o Enterprise Content Management.

Ci si soffermerà su alcune caratteristiche secondo noi innovative e avanzate di BEdita rispetto ad analoghi sistemi: la netta separazione fra backend di gestione e frontends applicativi, la sua natura di framework, il disegno a oggetti estendibile, l'uso di web application framework di nuova generazione, le specificità lato internazionalizzazione e la possibilità di creare relazioni semantiche libere.

Ci sarà una breve analisi comparativa fra ***BEdita*** *e un sistema di publishing Web2.0 che rappresenta lo stato dell'arte nel mondo del software libero:* ***Wordpress****. Si tratta di sistemi con scopi differenti la comparazione ha l'unico obiettivo di delineare meglio alcuni punti di forza e alcuni limiti e punti di debolezza di BEdita.,*

Saranno delineati anche i principali standard liberi di riferimento adottati nella progettazione del software: i classici standard W3C, Dublin Core, Sitemap, CMIS. Saranno poi descritte le componenti tecnologiche su cui ***BEdita*** *è costruito, un completo stack di software libero: PHP5, CakePHP, Smarty, MySQL5, jQuery.*

Saranno infine illustrati i principali campi di applicazione e una roadmap del progetto, tramite cui cercheremo di evidenziare l'aspetto semantico con l'introduzione, per esempio, di tecniche di Natural Language Processing.

Parole Chiave: CMS, Framework, AfferoGPL, Semantic Web, WebApplications

1 BEDITA: FRAMEWORK O CMS?

BEdita (pronunciato [bi'ɛdita]) è un framework per creare applicazioni Web pensato per il Web2.0, che ambisce a diventare uno strumento di riferimento per il prossimo Web3.0 e per il futuro Web Semantico. Il software, in realtà, non è stato pensato solo per il Web, ma è un framework per applicazioni da fruire *anche* su desktop/dispositivi mobili e per generare di stampa su carta (tramite l'integrazione con sistemi di Desktop Publishing).

Il software è stato sviluppato da due aziende italiane ChannelWeb S.r.l. e Chialab S.r.l., entrambe di Bologna, utilizzando uno stack di componenti unicamente di software libero. **BEdita** nasce da esigenze e requisiti raccolti da professionisti che lavorano su progetti e applicazioni Web da oltre dieci anni. Inizialmente è nato come strumento di supportto a uso *interno*: all'inizio del 2007 le due società hanno deciso dargli uno status indipendente procedendo a una completa re-ingegnerizzazione e riprogettazione e alla scelta di pubblicare il sofware con licenza *libera.* Un accordo legale regola la gestione del copyright condiviso tra le due aziende.

BEdita presenta molte analogie con i cosiddetti CMS (Content Management Systems) e spesso è presentato così per semplicità, ne rispetta infatti (almeno in parte) una definizione standard: *a tool that enables technical and non technical staff to create, edit, manage and finally publish a variety of content (such as text, graphics, video, documents etc).* Con la differenza che lato *publishing* non fornisce volutamente una soluzione default finita e univoca.

Una buona definizione di framework software ci può aiutare a capire meglio: a *framework is* ***incomplete****, though concrete,* ***driving solution*** *to recurring* ***high-value problem***. In questo senso BEdita è:

1. *incomplete* – non è uno software che da solo risolve un problema/bisogno di un utente; piuttosto è uno strumento per [web]designer/[web]developer che vogliono creare applicazioni frontend;
2. *driving solution to recurring high-value problem* – è utile in tutte le situazioni in cui creano applicazioni (*frontend*) dove è necessario gestire contenuti multimediali dinamici e complessi, e le loro relazioni semantiche; evita a un designer/developer di dover affrontare e risolvere tutte le tipiche problematiche correlate.

2 ARCHITETTURA E PRINCIPALI FEATURES

Gli elementi fondamentali del *semantic framework* sono due:

1. un'applicazione Web di *backend* per la redazione dei contenuti e delle loro proprietà e relazioni semantiche, con un'interfaccia utente innovativa per l'ergonomia tramite l'associazione cromatica fra tipologie di oggetti, l'uso estensivo del drag'n'drop e altre tecniche AJAX;
2. un API di frontend, servizi e specifiche per creare applicazioni frontend; con interfaccia Web in prima battuta, ma anche desktop/mobili sfruttando, per esempio, l'interfaccia nativa REST/ XML.

L'applicazione **backend** è una per ogni installazione del sistema o istanza, le applicazioni **frontend** sono costruite tramite un'**API** ed ereditano dal core del sistema **classi model e business logic**: possono essere poche linee di codice PHP o strutture più complesse. Questa separazione è una scelta voluta per ragioni di sicurezza, efficienza, scalabilità e per lasciare maggiore libertà possibile al lavoro del designer/developer.

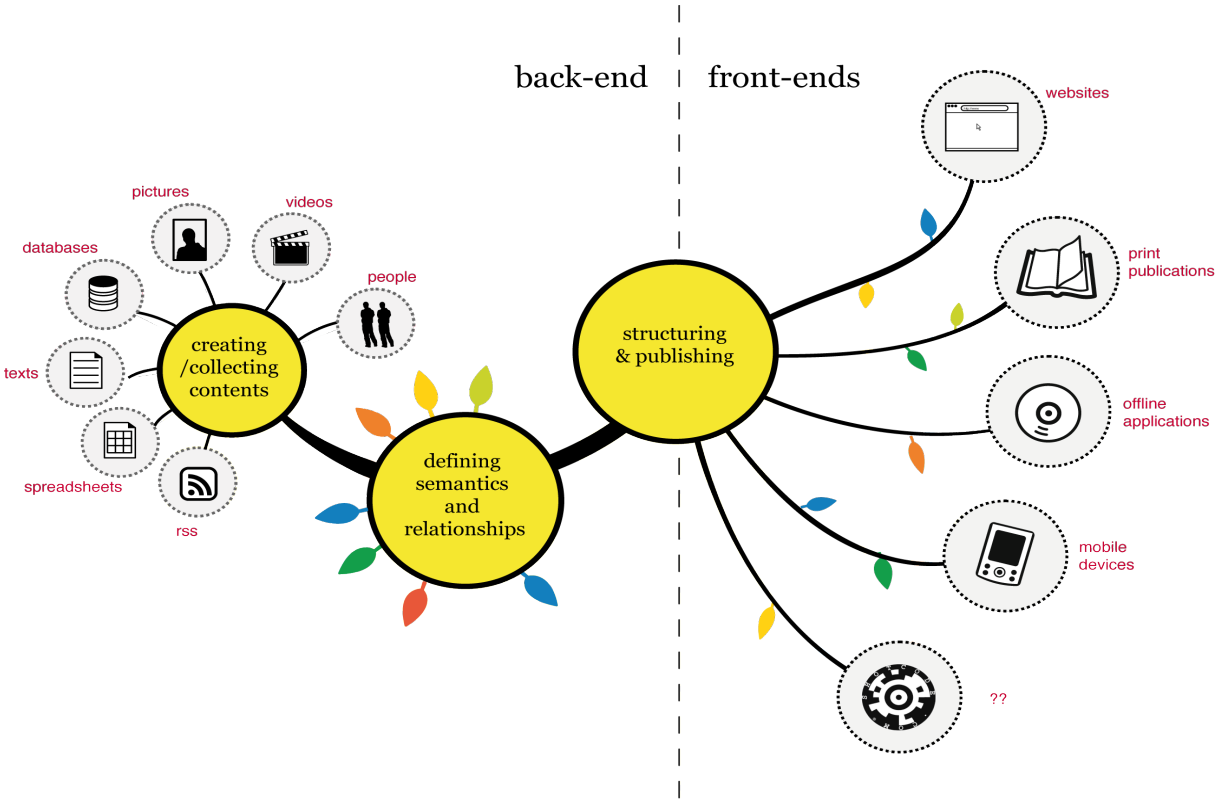

Figura 1. *Rappresentazione schematica delle attività lato backend e dei tipi di output di publishing lato frontend.*

Di seguito le caratteristiche/features più interessanti del sistema:

- **internazionalizzazione** completa: gestione di contenuti e interfaccia utente in qualunque lingua (tutte le lingue definite dallo standard ISO-693-3 sono in teoria supportate) – lato UI è utilizzato il formato gettext, mentre per oggetti/contenuti è stata creata una struttura dati dedicata;
- sistema a **oggetti**: ogni tipo di contenuto dentro BEdita è un oggetto, che può essere taggato, categorizzato, tradotto, geolocalizzato....; non solo documenti, ma anche immagini, video, eventi, schede anagrafiche, gallerie, newsletter...
- **estensione** oggetti: è possibile creare nuovi tipi di oggetti, aggiungere proprietà custom tipizzate agli oggetti stessi;
- **modularità**: moduli software specifici per task e per tipi di contenuti differenti;
- possibilità di creare **relazioni semantiche libere** fra oggetti;
- sistemi di **autenticazione** interno o esterno/pluggable tipo LDAP o OpenID;
- gestione integrata di sistema di **newsletter** (modulo dedicato);
- **statistiche** redazionali, integrazione di statistiche con sistemi come Google analytics, Piwik e log del web server (apache logs);
- **granularità** di accesso: definizione dinamica di gruppi di utenti con accesso in lettura/scrittura ai singoli moduli software;
- **interfaccia XML/REST** e **JSON** nativa per gli oggetti pubblicati;
- ogni oggetto ho un nome proprio univoco alfanumerico detto ***nickname*** da usare nelle URL e nei frontend per referenziarlo; è l'equivalente *semantico* dell'ID;

2.1 Punti di forza

Il mondo delle soluzioni di content management è estremamente vasto e multiforme, come si evince dai principali siti e directory dedicati all'argomento. **BEdita** prende spunti/idee da sistemi Web Content Management (WCM) basati su software libero come Drupal, Wordpress, Joomla, Plone verso

cui è in qualche modo *debitore,* ma è un progetto che nasce da esigenze e bisogni reali nella realizzazione di soluzioni Web da parte di professionisti, per cui ognuno di questi strumenti aveva delle limitazioni. Il Web è la principale interfaccia di BEdita, non l'unica: in vari contesti potrebbe sostituire uno dei WCM elencati sopra.

E' anche un Component Content Manager, cioè gestisce oggetti *riusabili*: un documento BEdita può contenere immagini/video/link che sono oggetti referenziabili in altri documenti, mentre lo stesso documento è associabile direttamente ad altre *pubblicazioni* gestite dalla stessa istanza. Ha elementi che lo possono avvicinarlo sistemi di Enterprise Content Management come Alfresco, di cui riportiamo una definizione autorevole: *strategies, methods and tools used to capture, manage, store, preserve, and deliver content and documents related to organizational processes*. Per cui possiamo sommariamente affermare che: *manage, store* e *deliver* sono a un buon livello, mentre *capture* e gli elementi *classici* di workflow necessari sono decisamente limitati, almeno per ora.

In definitiva è difficile trovare le giuste controparti per comparazioni puntuali, ma l'insieme dei punti seguenti crediamo rappresenti un set di caratteristiche unico, innovativo o quantomeno difficilmente rintracciabile:

- **separazione netta** fra backend (Web) e *applicazioni frontends (web, desktop, stampa)* – il primo pensato per un utente finale senza particolari competenze *tecnologiche*, mentre per i secondi (*frontends*) ci sono API, convenzioni e servizi per chi ha *competenze* (professionisti ICT, designer, developers);
- **chiara natura di framework**: vogliamo costruire applicazioni, non semplicemente visualizzare documenti dinamici; dal Web2 al Web3 con meno limitazioni possibili;
- **disegno a oggetti estendibile**: è possibile definire *nuovi tipi* di *model* (tramite API, plugin); estendere con attributi custom tipi di oggetti esistenti; tutto questo senza dovere per forza modificare lo schema del database, ma sfruttando il layer ORM;
- **riuso**: utilizzo di *web application framework* di nuova generazione (CakePHP), non creazione di web application frameworks *interni* (vedi **6. Third party**) da mantenere;
- **internazionalizzazione:** modulo dedicato per *traduttori;* riconoscimento automatico della lingua *migliore/preferita* per l'utente frontend (sfruttando dati dello UserAgent, scelte precedenti, geolocalizzazione);
- **relazioni semantiche libere:** è possibile definire *nuovi* tipi di relazioni semantiche tra tipi di oggetti; per esempio nel caso di un evento/presentazione pubblica: un *autore* [oggetto] *è relatore* [relazione] di un *evento* [oggetto] che *si svolge* [relazione] in un *luogo* [oggetto]; analogia concettuale con RDF del Semantic Web;
- alcune caratteristiche tipo *enterprise* (ECM) tipicamente non presenti su **architetture LAMP**, in un contesto dominato da piattaforme come .NET o Java EE;
- modulo **newsletter** integrato, funzionalità spesso mancante o presente con *addon.*

2.2 Confronto con Wordpress

Per chiarire alcune caratteristiche di **BEdita** può essere utile un confronto con un sistema di publishing Web2 di grande successo (meritato, a nostro parere) nel mondo del software libero: Wordpress, il cui utilizzo negli ultimi anni è letteralmente esploso.

Entrambi sono sistemi di publishing multiutente, con interfaccia utente grafica Web2 avanzata. Alcuni vantaggi e motivi per cui preferire BEdita:

- tanti *tipi* di oggetti/contenuti, ereditarietà – in Wordpress solo posts, media, links e commenti connessi fra loro (lato database), senza *disegno a oggetti* e altri tipi di relazioni possibili*;*
- sistema nativamente multi-publishing; possibile controllare e gestire più siti/pubblicazioni contemporaneamente dalla stessa istanza;
- struttura pubblicazione: è un albero di sezioni (*nodi*) dentro cui inserire contenuti (oggetti *tipo contenuto* - *foglie*) equivalente a directories e files di filesystem; su Wordpress ho *alberi* gerarchici di *pages* (non di altri tipi di contenuti come *posts*);
- i18n: ogni oggetto/contenuto può essere creato e tradotto in più lingue – lato frontend l'utente fruirà il contenuto automaticamente nella *lingua migliore*

Svantaggi o aspetti su cui Wordpress è, al momento, preferibile:

- alcune *features* di editing per accesso concorrente, versionamento (caratteristiche su cui si sta lavorando, disponibili nelle prossimi rilasci, vedi **10. Roadmap**)
- grande quantità di moduli e plugin di terze parti; filtri importazione/esportazione da/verso vari formati;
- documentazione online; grande ricchezza di risorse facilmente rintracciabili sul Web;
- sistema più semplice, più rapido e immediato, frontend già pronto e funzionale: il *target* è diverso, utente *meno esperto* non necessariamente designer o developer;

Il secondo e il terzo punto (moduli/plugin di terze parti, ricchezza di documentazione e risorse online) rappresentano obiettivi importanti nel lavoro di quest'anno che contiamo di ottenere con: migliore comunicazione online, partecipazioni a conferenze, coinvolgimento di terze parti e creazione di community attorno al progetto (vedi **9. Disseminazione**).

2.3 Third party

BEdita è stato pensato e disegnato con l'obiettivo di coniugare al meglio i concetti come riuso, efficienza e scalabilità. Indichiamo brevemente quali sono le componenti, i mattoni, su cui il sistema è stato costruito. Linguaggi, sistemi, librerie, frameworks: tutti software con *licenza libera*.

Requisiti di sistema:

- Architettura **LAMP**, Linux & Apache preferenziali e raccomandate, ma tutte le tecnologie usate sono cross-platform (quindi anche MacOSX, Windows, altri UNIX...)
- **MySQL5**: con uso di integrità referenziale, views, stored procedures; l'utilizzo di altri DBMS è *teoricamente* possibile, ma non supportato al momento;
- **PHP5**: utilizzo della sintassi a oggetti, uso estensivo delle eccezioni;

Componenti in bundle con il software distribuito:

- **CakePHP**: web application framework di nuova generazione, con impostazione molto simile al più famoso *RubyOnRails;* fornisce un'implementazione del pattern MVC, e supporto per i18n/l10n, Object-Relational Mapping, Unit Testing, DB migrations, sicurezza, caching, form validation;
- **Smarty**: libreria PHP di *templating* utilizzata per l'interfaccia utente (*View* del pattern MVC)*;*
- **jQuery**: libreria Javascript per interfacce AJAX, drag'n'drop, manipolazione DOM;
- molte altre librerie, fra cui citiamo phpThumb (manipolazione file multimediali) e TinyMCE (editor Rich Text);

2.4 Standard

Il progetto supporta e aderisce ad alcuni importanti standard **aperti** e **liberi**. Di seguito elenchiamo i principali:

- i *classici* standard W3C delle tecnologie Web
- Dublin Core per i metadati di oggetti/contenuti *editoriali*
- RSS/Atom per la pubblicazione automatica di feeds
- Sitemap: creazione automatica di *mappe* secondo il nuovo protocollo sitemap
- REST per l'interrogazione stile *WebServices* - SOAP per ora non è implementato;
- CMIS – Content Management Interoperability Services (*in lavorazione*) è un nuovo standard OASIS (attualmente ancora *draft*) di interoperabilità fra sistemi di gestione dei contenuti
- OpenDocument – creazione di filtri di importazione e esportazione (*in lavorazione*)

Altri standard più di basso livello da menzionare sono: ISO 639-3 per le lingue, UTF-8 come codifica a tutti i livelli.

3 CAMPI DI APPLICAZIONE

E' possibile delineare quattro campi o ambiti di applicazione dove riteniamo che BEdita sia più promettente, definendo quali possono essere gli attori/utilizzatori più interessati.

- **web *standard*:** pubblicazioni standard aziendali, servizi erogati da enti locali/PA, presentazione progetti, campagne, portali per grandi organizzazioni; attori: webagencies, agenzie di comunicazione, software house, system integrators;
- **editoria:** creazione di applicazioni verticali per l'editoria online (libri, riviste, quotidiani,..); implementazione di workflow redazionali dedicati; attori: operatori nel mondo dell'editoria;
- **soluzioni verticali specifiche:** individuazione di campi/settori specifici dove creare soluzioni *verticali e complete* dove ci sono spazi/mancanze; ad esempio nella gestione di eventi/festival/fiere; attori: aziende di consulenza e servizi ICT, startup;
- **servizi web** – per costruire nuove soluzioni e servizi per il Web2.0; attori: startup;

4 LICENSING

BEdita adotta un modello di business noto come *dual licensing*: è quindi disponibile con licenza libera e proprietaria.

La versione *libera* di **BEdita** è rilasciata con licenza **Affero GPLv3:** si tratta di una licenza relativamente nuova che è stata creata proprio nell'ambito dei software per servizi di rete (come nel nostro caso) per risolvere l'annoso problema dell'*ASP loophole.* Rispetto alla GPL classica cambia il concetto di *distribuzione* del software: non è necessario distribuire un pacchetto software, ma è sufficiente rendere disponibile un servizio via rete per ereditare diritti/doveri della GPL. Ad esempio: sono libero di costruire una qualunque applicazione usando software AGPL, ma sono tenuto rendere disponibile, con la stessa licenza, i sorgenti della mia applicazione una volta pubblicata su Internet.

Questo tipo di licenza è stata scelta proprio perché appare come una scelta nuova e efficace per trasferisce lo spirito e le intenzioni della GPL nell'ambito di servizi e applicazioni Web, un settore in forte espansione negli ultimi anni.

BEdita è anche disponibile in una versione con licenza proprietaria che contiene (attualmente) alcuni moduli *verticali* aggiuntivi (per *editoria* ed *e-commerce*).

La licenza libera scelta è quella che a nostro avviso meglio si adatta allo schema di *dual licensing* in quanto pone l'utente di fronte a una scelta netta e chiara:

- **scelgo la licenza AGPL**: ho tutti i diritti della licenza libera, posso usare il software a mio piacimento, ma non posso pubblicare applicazioni/sistemi su Internet senza rispettarne i *doveri* (tra cui quello di rendere disponibile il codice sorgente della mia applicazione);
- **detengo la licenza proprietaria**: posso pubblicare le mie applicazioni su Internet senza dover rispettare i *doveri* della AGPL, posso eventualmente creare applicazioni proprietarie;

Attraverso la licenza libera scelta è anche possibile immaginare un ciclo virtuoso di progetti software *liberi* basati sul framework, anche con chiari obiettivi *commerciali,* nel mondo della piccola e media impresa ICT. Un altro aspetto interessante correlato è la *condivisione* del copyright del progetto da parte di due piccole aziende operanti in ambiti complementari (consulenza e servizi ICT da una parte e comunicazione e editoria dall'altra): si tratta di un modello non semplice da adottare (è necessario un accordo legale tra le parti), ma che consente di realizzare prodotti più maturi e avanzati sfruttando al meglio le diverse competenze e professionalità disseminate nella piccola impresa (soprattutto nel contesto Italiano).

5 DISSEMINAZIONE

Non esiste al momento una vera community che lavora al progetto (di soggetti esterni alle due società), e l'utilizzo della versione Affero/GPL da parte di terzi è, al momento, molto limitato perché:

- lo sviluppo è stato fatto, volutamente, completamente *in house* per oltre 2 anni; scenario che potrebbe cambiare a seconda dei feedback/contributi che riceveremo;
- la pubblicazione della versione *libera* e l'attività di *disseminazione* è appena cominciata (pubblicazione su varie risorse nel Web, partecipazione a conferenze, recensioni);

Solo nel corso dei prossimi mesi, ragionevolmente entro la fine del 2009, saremo in grado di trarre delle conclusioni relativamente all'effettivo interesse: in caso di successo prevediamo che in maniera *naturale* evolveranno utilizzo e partecipazione al progetto.

6 ROADMAP

Per le attività future sul progetto è stata individuata una ***roadmap*** (*indicativa*) in tre punti, relativamente alla versione AGPL:

- giugno/luglio 2009 - rilascio prima versione *stabile*, bugfix, stabilizzazione attuali *features,* ottimizzazioni performance*;*
- autunno 2009 - rilascio versione 3.1 – miglioramento editing concorrente, web installer, caching e versionamento oggetti;
- primavera/estate 2010 – rilascio versione 3.2 - estensione del sistema di permessi sugli oggetti; introduzione di tecniche di NLP (Natural Language Processing) per categorizzazione e associazione *automatica* di relazioni semantiche.

Riferimenti

http://www.bedita.com sito di riferimento del progetto, in particolare http://www.bedita.com/beopen informazioni e documentazione sulla versione libera.

http://www.w3.org/2001/sw/ sito di riferimento del progetto Semantic Web al W3C

http://www.aiim.org Association for Information and Image Management o AIIM, associazione internazionale per formazione, ricerca e best-practices nel campo dell'information management e in particolare dei sistemi ECM

http://www.cmswatch.com/ portale informativo dedicato al mondo dei CMS

http://www.oss-watch.ac.uk/resources/duallicence2.xml articolo su OSS Watch dedicato al Dual Licensing

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cmis standard di interoperabilità CMIS in via di definizione presso OASIS

http://cakephp.org portale ufficiale del Web Application Framework CakePHP