PENGEMBANGAN APLIKASI EXTENDABLE CONTENT MANAGEMENT SYSTEM

Adi Wibowo[1)], Liliana[2)] , Claffyan Wicaksono[3)]
[1,2,3)]Program Studi Teknik Informatika Universitas Kristen Petra Surabaya
Jl. Siwalankerto 121-131 Surabaya Telp (031)-2983455
e-mail : [1)] adiw@petra.ac.id, [2)] lilian@petra.ac.id

Abstrak
Content Management System (CMS) adalah aplikasi untuk membuat sebuah website dengan mudah dimana editor tidak diharuskan untuk memahami HTML. Pada saat ini terdapat beberapa CMS yang terkenal, yaitu Wordpress, dan Drupal. CMS-CMS tersebut memiliki kelebihan yaitu dapat ditambah kemampuannya menggunakan extension. Bila CMS sebelumnya tidak dapat menerima komentar di setiap halamannya, extension memungkinkan pengguna menambah dan melihat komentar di setiap halaman website. Demikian juga extension dapat menambah kemampuan mengisi dan melihat galeri gambar pada website tersebut.Kedua CMS di atas memiliki kelebihan yaitu jumlah extension yang banyak. Tetapi wordpress memiliki kekurangan, yaitu blok-blok yang membangun tampilan website tidak dapat dipilih oleh editor. Drupal memberi pilihan pada editor blok-blok apa yang ingin ditampilkan di website, tetapi konsep node di drupal cukup kompleks terutama bila dibutuhkan hanya website yang sederhana. Untuk itu penelitian ini menyarankan adanya sebuah CMS yang mudah dikembangkan menggunakan extension, tetapi memberi kesempatan pada editor memilih blok mana yang ingin ditampilkan, dan mudah digunakan.Penelitian ini menghasilkan konsep penambahan extension otomatis, dimana CMS dapat mengenali secara otomatis extension yang ditambahkan, membuat tabel-tabel yang diperlukan extension secara otomatis, dan menggunakan konsep hook dimana CMS akan memanggil extension hanya bila diperlukan. Extension yang dikembangkan sebagai contoh pada penelitian ini adalah komentar, dan galeri gambar.
***Kata Kunci** : Extension, extension, content management system*

1. PENDAHULUAN

Dengan meningkatnya kebutuhan organisasi untuk menyediakan informasi baik bagi pengguna internal, maupun eksternal organsasi membutuhkan sistem yang mendukung penyediaan informasi tersebut. Sistem yang sering digunakan adalah Content Management System (CMS). CMS adalah aplikasi untuk membuat sebuah website dengan mudah dimana editor tidak diharuskan untuk memahami HTML. Pada saat ini terdapat dua CMS yang banyak digunakan oleh organisasi. Dari 10.000 website paling ramai dikunjungi 52,61% menggunakan Wordpress, posisi dua Drupal digunakan oleh 19,67% organisasi (Builtwith, 2011).

Kedua CMS tersebut memiliki kelebihan dan kekurangan masing-masing. Drupal memiliki konsep node yang sangat fleksibel, tetapi membuatnya menjadi susah dikuasai. Organisasi yang menggunakan Drupal harus memperkerjakan programmer khusus untuk menyesuaikan Drupal sesuai kebutuhan organisasi tersebut. Kelebihannya adalah semua jenis kebutuhan penyediaan informasi dapat disediakan oleh Drupal. Wordpress sangat mudah digunakan dan dikembangkan, tetapi karena berkembang dari konsep blog, maka kemampuan Wordpress dalam penyajian informasi yang beragam tidak sebaik Drupal (Rackspace, 2013). Wordpress juga memiliki kekurangan, yaitu blok-blok yang membangun tampilan website tidak dapat dipilih oleh editor. Drupal memberi pilihan pada editor blok-blok apa yang ingin ditampilkan di website, tetapi konsep node di drupal cukup kompleks terutama bila dibutuhkan hanya website yang sederhana.

Kedua CMS dapat dikembangkan kemampuannya lebih lanjut menggunakan extension. Extension memberikan kemampuan tambahan, misalnya filter isi, penerjemahan isi website, pembuatan menu, penambahan kalender, hingga extension yang kompleks seperti fasilitas e-commerce.

Penelitian ini bertujuan mempelajari sistem pengembangan CMS yang dapat ditambah kemampuannya menggunakan extension. CMS yang dikembangkan juga harus dapat memberi kesempatan pada editor memilih blok mana yang ingin ditampilkan, dan mudah digunakan.

2. TINJAUAN PUSTAKA

Menurut Bernard Kohen (2010), Content Management System (CMS) adalah sebuah aplikasi biasanya berbasis web yang menyediakan kemampuan bagi beberapa pengguna, dengan hak-hak akses yang berbeda-beda, untuk mengelola isi, data, atau informasi yang disediakan oleh proyek website, atau aplikasi internet / intranet.

Pengelolaan isi meliputi proses pembuatan, pengubahan, pengarsipan, publikasi, kolaborasi, pelaporan, dan distribusi isi, data, dan informasi melalui website.

Sebagai contoh CMS adalah aplikasi yang memberikan kemampuan untuk:

1. Membuat, mengubah, mempublikasi, mengarsip halaman-halaman web, artikel, blog
2. Membuat, mengubah acara di kalender acara
3. Menambah, mengubah produk di gudang, deskripsi, spesifikasi produk, harga, foto, dll.
4. Memasukkan, mengubah, melihat atau mencetak slip pengepakan, dan invoice
5. Membuat laporan, dan data statistik
6. Menambah, mengubah, dan memberi hak akses bagi editor website.

Pada penelitian ini yang dimaksud dengan CMS adalah aplikasi yang memenuhi fungsi-fungsi pertama, dan kedua saja. Penelitian ini tidak bermaksud membuat CMS secara utuh melainkan konsep extension dan block dalam sebuah CMS.

Menurut Williams, B., Richard, O. & Tadlock, J. (2011), WordPress Extension adalah sebuah aplikasi yang berfungsi menambahkan serangkaian fitur atau layanan tertentu pada weblog WordPress. Extension pada WordPress disimpan pada wp-content/plugins. Setiap extension akan dijalankan melalui konsep hook. WordPress berisi dua hook yang berbeda, yaitu *Action* dan *Filter*. *Action hook* memungkinkan Wordpress untuk menjalankan (*trigger*) kode extension yang pada titik-titik tertentu selama pembuatan halaman website. Sebagai contoh, Wordpress dapat memicu fungsi khusus untuk dijalankan setelah pengguna mendaftarkankan sebuah akun di WordPress, atau menjalankan extension saat sebuah widget diaktifkan. *Filter hook* digunakan untuk memodifikasi teks sebelum menambahkan atau mengambil dari database. Contoh hook di Wordpress ditunjukkan pada Gambar 1.

```
<?php add_action(  'wp_footer', 'boj_example_footer_message', 100);
function boj_example_footer_message ( )   {
      echo  'This site is built using <a href="http://wordpress.org"
     title="WordPress publishing platform">WordPress</a>.';
} ?>
```

Gambar 1. *Action Hook* dari Wordpress

Hook adalah sebuah sistem dimana CMS akan mendaftar seluruh event yang terjadi selama proses pembuatan sebuah halaman, dan menyediakan slot dimana extension dapat mendaftar untuk diaktifkan bila event tersebut terjadi. Sebagai contoh, bila ada event pembuatan menu, maka CMS dapat mengaktifkan extension-extension yang berhubungan dengan menu, misalnya extension penerjemahan menu.

Menurut VanDyk, J. K. (2008) Drupal dibangun di atas sistem hook, kadang-kadang disebut callback. Selama eksekusi, Drupal mengecek pada modul-modul yang ada jika modul-modul tersebut ingin menjalankan sesuatu. Sebagai contoh, ketika sebuah node sedang dimuat dari database sebelum ditampilkan pada halaman, Drupal memeriksa semua modul yang diaktifkan untuk melihat apakah mereka ada yang menerapkan fungsi hook_node_load(). Jika demikian, hook dari Drupal akan mengeksekusi modul sebelum render node pada halaman.

Modul Drupal dapat berfungsi pada lokasi mana saja selama masih di tempat Drupal mencari modul - modul seperti sites/all/modules. Untuk mengelompokkan modul - modul pada Drupal, dapat dibuat suatu sub-folder seperti sites/all/modules/custom. Bentuk implementasi hook di Drupal ditunjukkan pada Gambar 2.

```
        /* Implementation of hook_menu ().  */
              function annotate_menu() {
                  $item['admin/config/annotate']=array(
                     'title' => 'Node annotation',
                     'description' => 'Adjust node annotation options.',
                     'position' => 'right',
                     'weight' => -5
                     'page callback' => 'system_admin_menu_block_page',
                     'access arguments' => array('administer site configuration'),
                     'file' => 'system.admin.inc',
                     'file path' => drupal_get_path('module','system'),
                  );
                  $items['admin/config/annotate/settings'] = array(
                     'title' => 'Annotation settings',
                     'description' => 'Change how annotations behave.',
                     'page callback' => 'drupal_get_form',
```

```
                    'page arguments' => array('annotate_admin_settings'),
                    'access arguments' => array('administer site configuration'),
                    'type' => MENU_NORMAL_ITEM,
                    'file' => 'annotate.admin.inc',
                );
                return $items;
        }
```

Gambar 2. *Class Diagram* dari CMS

3. METODE PENELITIAN

Penelitian dimulai dengan membandingkan implementasi extension di Wordpress dan Drupal. Hasil dari pembandingan akan digunakan untuk menyusun konsep implementasi extension. Setelah konsep selesai dibuat, diuji dengan diimplementasikan ke beberapa class di bahasa pemrograman PHP. Implementasi extension pada penelitian ini diujicobakan pada tiga extension, yaitu comment, photo gallery, dan calendar.

4. HASIL DAN PEMBAHASAN

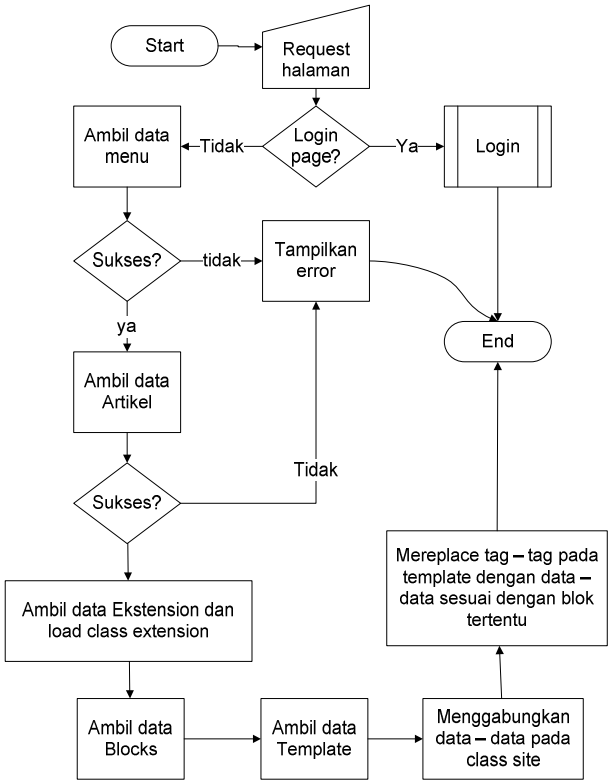
Kedua CMS di atas menggunakan konsep hook untuk mengetahui kapan sebuah extension harus dijalankan oleh CMS. Setiap extension harus memberitahu pada hook-hook mana extension dapat diaktfikan. Baik di wordpress, maupun drupal extension harus memberitahu ke CMS hook mana yang ingin didaftarkan melalui fungsi. Programmer harus membuat sebuah fungsi di PHP untuk memberitahu hook-hook yang digunakan oleh extension tersebut. Konsep hook sangat baik diimplementasikan karena memberikan fleksibilitas kapan extension dapat diaktifkan. Pada penelitian ini hook diimplementasikan secara terbatas, yaitu pada pembuatan block saja. Untuk memilih hook dimana extension diaktifkan digunakan variabel dalam extension tersebut, tetapi editor dapat mengubahnya melalui menu dalam CMS.

Block adalah area yang membentuk sebuah halaman website. Block sudah ditentukan sebelumnya terdiri atas header, header_child, menu_child, content, sidebar_left, sidebar_right, footer. Setiap block dibuat menggunakan template dengan nama file <nama_block>.blk. Dalam setiap block terdapat *template variable*. *Template variable* adalah bagian dari template yang dapat diganti sesuai dengan kebutuhan halaman website tersebut. Sebagai contoh pada block sidebar_right terdapat dua variabel, yaitu {%sidebar_right_title%}, dan {%sidebar_right_content%}. Setiap block dapat diisi oleh berbagai jenis isi. Block header dapat diisi dengan menu navigasi, atau extension. Block sidebar dapat diisi dengan teks bebas, atau extension. Block content akan diisi dengan isi artikel dan juga extension.

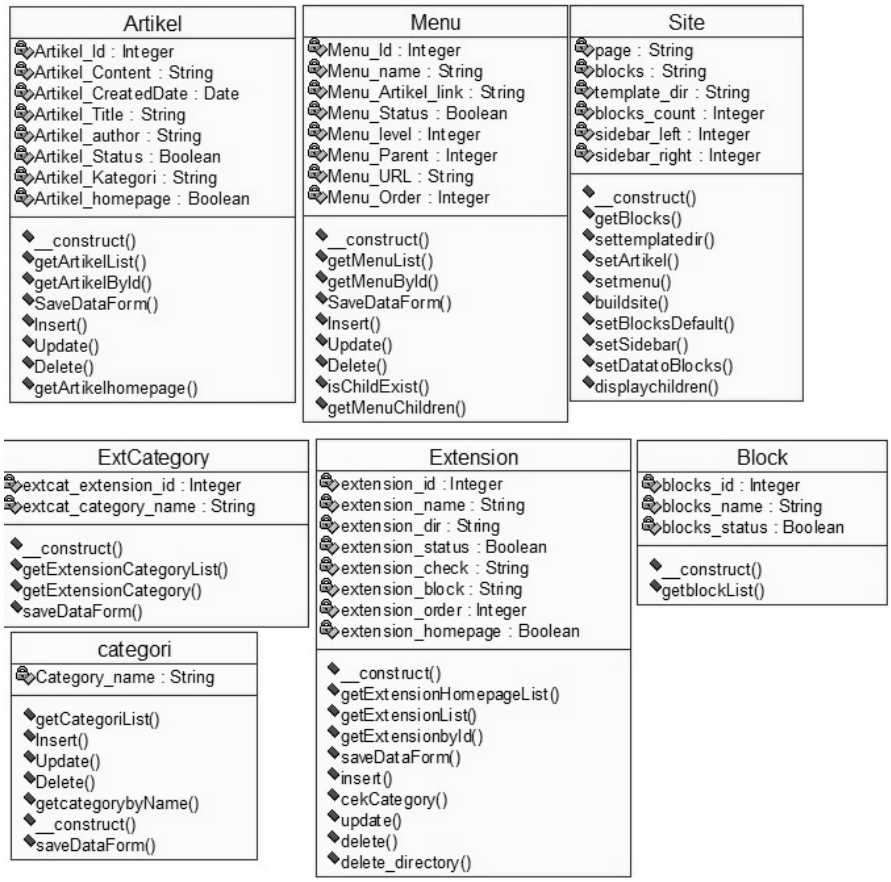
Secara umum proses pembuatan setiap halaman website oleh CMS ditunjukkan pada Gambar 3. Ketika pengunjung website meminta sebuah halaman dari CMS, CMS akan memeriksa apakah pengunjung meminta halaman login atau isi website. Bila pengunjung meminta halaman website, maka CMS akan mengambil data menu dari database, kemudian mengambil isi halaman (data artikel), membuat extension block beserta isinya, menggabungkan semua unsur pengisi sebuah halaman (menu, artikel, block, extension) menjadi satu dan mengirimkannya ke pengguna (*browser*).

Extension memiliki standar yaitu harus minimal memiliki file index.php, dan instal.php. File instal.php adalah file yang dijalankan ketika extension ditambahkan ke CMS. Pada file instal.php diharuskan ada variabel $default_block yang berisi akan block awal tempat *extension* tersebut akan ditampilkan. Di dalam file index.php harus ada variabel $dataextend yang terdiri dari $dataextend['title'] yang berisi dari judul dari *extension* dan $dataextend['content'] yang berisi hasil yang akan ditampilkan ketika index.php dipanggil pada halaman front end. File uninstal.php dapat ditambahkan pada *extension* apabila ketika *extension* dihapus dari CMS terdapat proses yang perlu dilakukan, misalnya menghapus tabel yang digunakan oleh *extension*. File index.php, instal.php, dan uninstal.php harus berada pada direktori terluar di dalam *extension package* yang berupa file zip.
Untuk menambah sebuah extension baru, administrator dapat mengunggah *extension package* melalui form Add Extension. *Extension package* akan diekstrak ke direktori *module*. Apabila terdapat file dengan nama instal.php, maka file tersebut akan dijalankan segera setelah file.zip selesai diekstrak. Bila menu Delete Extension di form Administrasi Extension dijalankan, CMS akan menghapus direktori extension beserta file-file dari extension tersebut. Sebelum proses penghapusan dilakukan file uninstal.php yang terdapat pada direktori extension akan dijalankan lebih dulu.

Extension dapat diatur agar hanya tampil di block-block di dalam artikel dengan kategori tertentu. Misalnya untuk artikel kategori Berita, block sidebar dapat diisi dengan *news extension*. Sedangkan untuk artikel kategori Umum, block sidebar dapat diisi dengan *comment extension*. *Class diagram* dibuat untuk mengimplementasikan konsep rekonstruksi halaman website seperti ditunjukkan pada Gambar 4.



Gambar 3. Proses Rekonstruksi Halaman Website



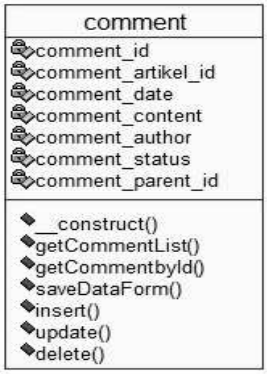
Gambar 4. *Class Diagram* dari CMS

```
    $artikelId = (int)$_GET['artikelId'];
    $data = Artikel::getArtikelbyId($artikelId);
    $situs = new site($template_path);
    $situs->setArtikel($data);
    $data_extcat = ExtCategory::getExtensionCategoryList($data->artikel_category);

    foreach($data_extcat['hasil'] as $extcat){
        $ext_include_id = $extcat->extcat_extension_id;
        $ext = Extension::getExtensionbyId($ext_include_id);
        $file_include = $ext->extension_dir . "index.php";

        $extension_block = $ext->extension_block;
        include_once $file_include;
        $situs->setDatatoBlocks($dataextend, $extension_block);
    }
    $situs->buildsite();
```

Gambar 5. Implementasi Pembuatan Halaman Website

Pada Gambar 5 terlihat proses pembuatan sebuah halaman website. Setiap request dari pengguna mengandung ID dari artikel yang diminta. CMS akan mengambil isi artikel sesuai ID tersebut. CMS kemudian membuat obyek Situs dan mengisi data artikel sesuai isi artikel yang telah diambil sebelumnya. CMS mengambil semua extension yang aktif untuk artikel tersebut, kemudian mengisi block-block artikel sesuai konfigurasi entension.

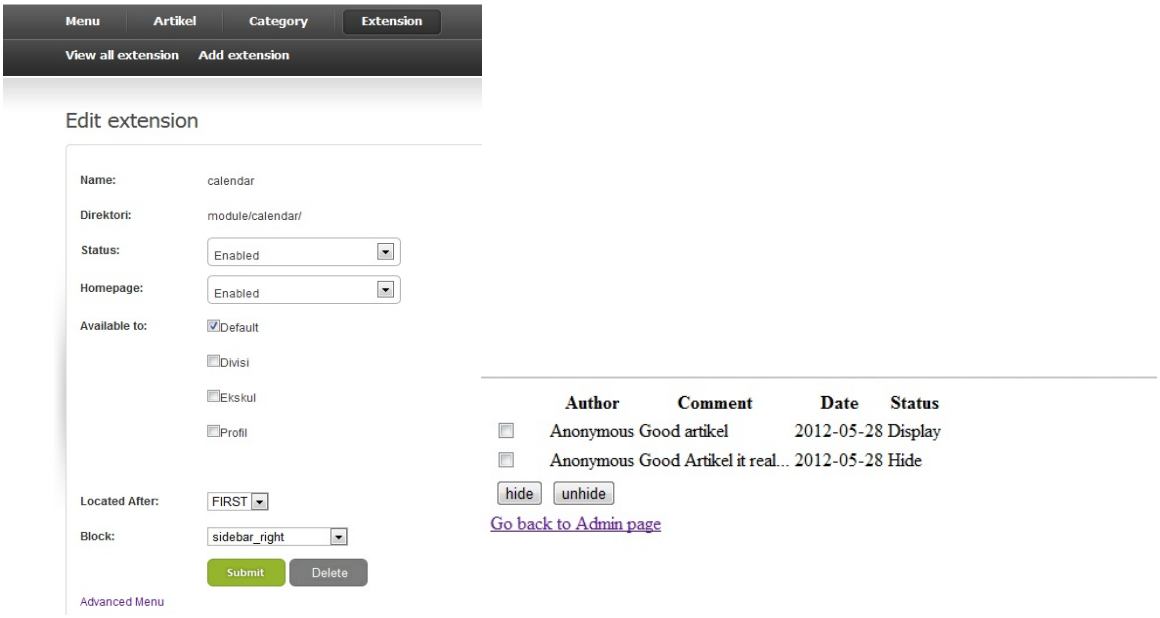
Untuk menguji implementasi extension pada penelitian ini dibuat extension untuk comment, calendar, dan image_gallery. Implementasi dari *comment extension* ditunjukkan pada Gambar 6.



Gambar 6. *Class Diagram* dari *Comment Extension*

Pada class Comment terdapat fungsi getCommentList() yang bertugas mengambil seluruh isi komentar dari sebuah artikel. Hasil dari fungsi tersebut disimpan di variabel $dataextend['content']. Variabel $dataextend['content'] digunakan oleh prosedur pada Gambar 5 untuk mengisi block dari sebuah halaman website.

Bila sebuah extension sudah ditambahkan dan diaktifkan di CMS, operasi edit extension ditunjukkan pada Gambar 7a. Setiap extension dapat memiliki halaman administrasi untuk mengelola data setiap extension. CMS akan memeriksa apakah extension memiliki file admin.php. Bila ada, maka menu administrasi extension dapat diakses oleh editor. Form administrasi *comment extension* ditampilkan di gambar 7b.



(a) (b)

Gambar 7. User Interface dari Administrasi Comment Extension

Hasil implementasi menunjukkan bahwa pengembangan konsep CMS dapat diterapkan. Kekurangan yang perlu diatasi pada penelitian selanjutnya adalah jenis-jenis hook yang lebih banyak dibandingkan hanya proses pembuatan block seperti pada penelitian ini.

5. KESIMPULAN

Penelitian ini mengusulkan konsep extension untuk CMS menggunakan hook untuk event pembuatan block. Proses aktifasi extension melalui file index.php yang harus ada di dalam setiap extension. Hasil implementasi menunjukkan konsep dapat diterapkan, dan perlu dikembangkan untuk jenis-jenis hook yang lain.

DAFTAR PUSTAKA

Bernard Kohen, 2010, *What is a Content Management System (CMS)? : Content Management System (CMS) and other spin-off terms definition(s)*, http://www.comentum.com/what-is-cms-content-management-system.html, diakses pada tanggal 1 Mei 2013

Builtwith, (2011), *Content Management System Distribution*, http://trends.builtwith.com/cms, diakses pada 1 Mei 2013.

Rackspace Support, 2013, *CMS Comparison: Drupal, Joomla and Wordpress*, http://www.rackspace.com/knowledge_center/article/cms-comparison-drupal-joomla-and-wordpress, diakses pada 1 Mei 2013

VanDyk, J. K. (2008). *Pro Drupal development*. United States of America: Apress.

Williams, B., Richard, O. & Tadlock, J. (2011). Professional WordPress extension development. United States of America: John Wiley & Sons.