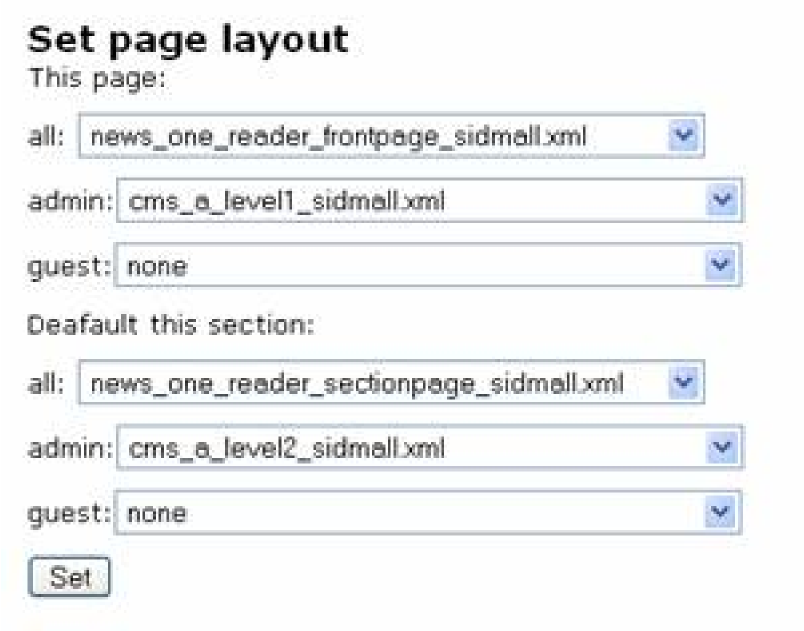
Sitemap configuration

Firstly, shapeCMS uses the authentication scheme. It is used by the load_page_template.xq xquery, to append an admin interface for page layout selection, to all pages for admin accounts. The page layout interface looks like below, allowing you to set different page templates for users with different roles (admin, and guest, in this example).



Set page layout

This page:

all: news_one_reader_frontpage_sidmall.xml

admin: cms_a_level1_sidmall.xml

guest: none

Deafault this section:

all: news_one_reader_sectionpage_sidmall.xml

admin: cms_a_level2_sidmall.xml

guest: none

Below, I show a part of the page layout metadata scheme for news_one. It can contain information on page layout files to use for certain levels <default>, and for certain pages <url>.

```
<?xml version="1.0" encoding="UTF-8"?><pagelayouts>    <pages>
<default>           <html>
<admin>cms_a_level2_sidmall.xml</admin>
<guest>none</guest>
<all>news_one_reader_sectionpage_sidmall.xml</all>           </html>
</default>       <base>../</base>       <page>           <url>.</url>
<pattern>                <html>
<admin>cms_a_level1_sidmall.xml</admin>
<guest>none</guest>
<all>news_one_reader_frontpage_sidmall.xml</all>               </html>
</pattern>       </page>       <page>
```

Fixed page layout schemes and content location metadata

The following is an example of a sitemap configuration with a fixed layout scheme metadata xml file and a fixed content location metadata file.

The first part of the match pattern (weblog) can be used to set the discussion and page layout schemes (see the next example). Then a number of * sets for what page depth this is for. You can add as many levels as you like, by adding more matchers in the site map.

```
<map:match pattern="weblog/*/*/*.htm">
```

The Cocoon authentication scheme is used.

```
        <map:act type="auth-loggedIn">                <map:parameter name="handler"
value="demohandler"/>                <map:act type="auth-protect">
<map:parameter name="handler" value="demohandler"/>
<map:generate src="xq/load_page.xq" type="xquery">
<map:parameter name="handler" value="demohandler"/>
```

ID is used during content creation, to add creator metadata to content items. The ID is taken from the cocoon authentication scheme.

```
                              <map:parameter name="ID" value="{ID}"/>
```

Role is used to select the page layout template (see above). It is also taken from the Cocoon authentication scheme.

```
                              <map:parameter name="role" value="{role}"/>
```

This is the default layout, which will be used, if you set no layout using the web interface (see above)

```
                              <map:parameter name="default" value="blogcomment.xml"/>
```

This parameter can be used together with pdf generation. It then replaces references to this.pdf with the page stated here. (not yet fully implemented in the sitemap, but the scripts work)

```
                              <map:parameter name="sida" value="{../../3}"/>
```

Base, is simply the first part of the URL. It is stated here, since it is used for hyperlinks, and can differ both from the discussion and pagelayout parameters.

```
                              <map:parameter name="base" value="weblog/"/>
```

The imageprefix is put before each URL in the page, to links within the current site.

```
                              <map:parameter name="imageprefix" value="../../../"/>
```

This is the name of the content location hierarchy. It is shared between weblog and forum, in the default configuration. You can add yet another pipeline that shares it, if you want to. This gives the same contents to both layouts. In the content metadata it is matched against the tag <location><live><at><combo>, when contents are retrieved from the database. The location hierarchy is used to describe what pages exist, at a given location. That can be used to prevent some user categories to publish materials at new pages, while allowing them to publish materials at existing pages.

```
                              <map:parameter name="discussion" value="combo"/>
```

This is the name of the layout scheme for the weblog. The forum has a different layout scheme. In the scheme, you can assign page layouts, and section layouts, for different roles. You can have as many layouts as you want, for one content location hierarchy. The layout scheme stores all the decisions you make about page layouts, using the web interface above.

```
                              <map:parameter name="pagelayout" value="comboweblog"/>
```

This states what metadata to use for creating the URL of new sections. You can use “byTitle”, in which case the title is used. You can also use byUrl, in which case the URL metadata is used. (you have to provide these fields during creation of a section, of course)

```
                              <map:parameter name="createDiscussion" value="byTitle"/>
```

The URL is the rest of the URL sent by the browser, except for the first part (weblog). That is stored in the page metadata. (<location><live><at><combo><page>stens blog/discs/bobdylan</page>). Of course, several locations can be stored in the metadata for a particular item. The item is not actually stored at that physical location.

```
                              <map:parameter name="url"
value="{../../1}/{../../2}/{../../3}"/>                    </map:generate>
```

This transformation uses Xinclude, to put the page together, based on the page layout retrieved above.

```
                    <map:transform type="xinclude"/>                    <map:serialize
type="html"/>            </map:act>      </map:act></map:match>
```

Dynamic page layout schemes and content location metadata

You can also include a pipeline that matches all pages, which are not defined elsewhere. Then, the first part of the match is used to determine which page layout scheme and location metadata to use. If you go to a “new” page, by default, this sitemap defines that a 404 message page should be displayed. But if you are logged in as Admin, you also get the web interface for selecting layouts. You can then define page layouts for different roles, like for other sections. Page layout definitions and site structure metadata is then created. If you set a content management page layout for some role, you can then begin to create contents in the new section.

```
<map:match pattern="*/*.htm">       <map:act type="auth-loggedIn">
<map:parameter name="handler" value="demohandler"/>          <!-- Nödvändigt att
göra auth-protect också, för att få tag på parametern ID från användarlistan
-->            <map:act type="auth-protect">                    <map:parameter
name="handler" value="demohandler"/>                    <map:generate
src="xq/load_page.xq" type="xquery">                         <map:parameter
name="handler" value="demohandler"/>                          <map:parameter
name="ID" value="{ID}"/>                         <map:parameter name="role"
value="{role}"/>                         <map:parameter name="default"
value="404.xml"/>                         <map:parameter name="sida"
value="{../../2}"/>                         <map:parameter name="imageprefix"
value="../"/>
```

As you see, the location metadata (discussion) and pagelayout schemes to use, are here defined by the first part of the matcher. This is the main difference between the fixed and dynamic layouts. Then the base is also the same as these two.

```
                              <map:parameter name="discussion"
value="{../../1}"/>                         <map:parameter name="pagelayout"
value="{../../1}"/>                         <map:parameter name="base"
value="{../../1}/"/>                         <map:parameter name="createDiscussion"
value="byUrl"/>                         <map:parameter name="url"
value="{../../2}"/>                    </map:generate>                    <map:transform
type="xinclude"/>                    <map:serialize type="html"/>
</map:act>      </map:act></map:match>
```

(Location) Metadata

Metadata are structured in metadata xml files. This allows you to dynamically create metadata, if you want to allow any metadata to be used. More importantly, it allows you to restrict possible metadata input, to for instance the categories in NITF or some other controlled vocabulary. Location metadata is simply a metadata file which is used to define a location. Therefore, it is a good idea to use the prefix meta_ (for instance) for metadata, to keep them apart. This also means that you can use any metadata file as a location definition, and that you can suddenly do that, even if you did not plan to do that from the outset. In this way, you could define a browsing interface for all contents using a specific metadata set. Or, you could swiftly re-organize your entire site, to follow a new metadata structure.

Below, you can see a part of the metadata structure for the news one headlines. This means that marking certain contents as appearing in some headlines section, is just to add metadata. It also means that a web interface can be defined to manage the headlines, using the metadata as a location. Also the news alpha sample uses this metadata file.

```
<?xml version="1.0" encoding="UTF-8"?><conferences>    <meta>
<last>1</last>        <base>news_one_headlines</base>
<conference>news_one_headlines</conference>    </meta>    <section>
<title>headlines</title>        <no>1</no>        <id>headlines</id>
<date>2005-12-03T17:25:02.343+01:00</date>        <live>yes</live>
<cutBranches/>        <ip/>        <creator>cocoon</creator>        <meta>
<last>0</last>        </meta>        <section>            <title>wind</title>
<no>1.0</no>            <id>headlines/wind</id>
<date>2005-12-03T17:25:02.343+01:00</date>            <live>yes</live>
<cutBranches/>            <ip/>            <creator>cocoon</creator>
<meta>                <last>0</last>            </meta>        </section>        <section>
<title>earth</title>            <no>1.0</no>            <id>headlines/earth</id>
<date>2005-12-03T17:26:07.062+01:00</date>            <live>yes</live>
<cutBranches/>            <ip/>            <creator>cocoon</creator>
<meta>                <last>0</last>            </meta>        </section></conferences>
```