**Tru-Medical: design alternative #2 / Working Prototype**



**1. footer on all pages** ( in application.html.erb )

**2. catelog request:**

1. instead of having category request create a new user, have it create a new catelog\_request record. It should also check to see if this is a user we already know about, and if so, link the catelog request record to that user, assocation called 'catelog\_requests'?

2. what happens after creating a new catelog\_request? right now it is trying to render views/users/show.html. Want it to give the user a quick confirmation that his request has been received, and will be mailed shortly. implies sending the form in using ajax; putting confirmation msg in the popup, and then having the popup close, or saying 'To close and return to ... click x" . See code created for AJAX-based variant of 'Sign Up' form.

3. expand the form to include a choice: does he want printed copy/copies of the catelog mailed to him, or would he like a pdf copy emailed to him? Maybe a 'click here to download a pdf copy' / fill in the form below if you would like us to mail you a physical copy: in which case we'll need his physical mailing address — something we are not capturing now.

4. try other form builders: formtastic, its newer cousin: simple\_form

5. check to see if anyone else thinks its odd to have to create a dummy @user in order to pass it to the form\_for when the form is going to be used to create an instance of the type. Trick is to create those users in memory, but not save them to the database unless the user actually fills out and submits the form.

**3. search for Sammons part number equivalent**

1. How going to do this?

2. Once you find it what? : go to page for the equivalent part in TruMedical catelog? What if the match is not exact?

**4. header on all pages (actually fields on the upper right )**

1. register/login: build own using Authlogic Rails plugin (or whatever used on aaox); want stuff to come up in popups above the home page, not go off to separate pages — and especially not the slow netsuite-generated ones.

**5. 2nd, 3rd level category navigation**

1. update the category\_tree\_v3.xls file to add (i) the new images, (ii) the 3rd level categories. [DONE] 12/10

NOTE: Although this is a tree with a maximum depth of three, it does not go uniformly down to level 3 in all branches. Many go only to level-2. Force it to 3 by adding a dummy 3rd level 'All' category ito each 2nd level node that does not have 3rd level chldren as did in VMS? No, in that case, it was only a very small % of the 2nd level categories that required a dummy third level category 'All'. Here, well over half of the categories bottom out at level 2. Time for n-level category tree software. Will need both a nav-HTML builder and menu ( and/or category landing pages ). In this case I've hand-built the left-nav so the .rb script that builds the HTML for the left-name won't be necessary.

2. Make sure that we have all of the 'cateogry images' referenced in the excel file. Make sure they have been resized (to 120x80 jpgs), and are in the correct directory, eg.., /app/images/categories [DONE] 12/10

3. Reload the categories table in the database so that the categories pages will pull in these images. [DONE] 12/10

4. Restart the application and dbms servers, bring up a browser against the local version of the application, see how the category landing pages look. [DONE]

5. Fix anything that is breaking —

— Modalities

references to:

710025-000 (Accessories ) [ image file missing

710016-000 (Combination) [ image file missing ]

710054-00 (Electrodes) [ short a digit in the filename ]

710065-000 (Lotions and Gels) [ image file missing ]

problem: /public/images/categories contains only the thumbnails made from the -XXX files Bill sent; it does not contain the 710016-000 files.

fix: Find them in /public/images/[products, and make 120x80 thumbnails from them. Place the thumbnails in /public/images/categories. Do this for /images/products/modalities for now. Batch resize all of the files in that directory, copying them into /public/images/categories. Then go through /images/categories and delete all but the -XXX and -000 files.

problem: leaf-level category (modalities/clinical electrotherapy/ultrasound) is showing several images of e.g., Dynatron 125. Should it be? Yes: these are all of the individual products in this leaf-level category. Image of Dynatron Intellect Transport 710032-000 is missing. Why? Have 710031-00, 710033-000 and 710034-000 but not 710032-000. Why not?

6. insert the images for all of the top level product categories so that if Bill shows the Working Prototype to his partners, the left nav will work across the full product tree. This will also let us surface any 'missing-image' issues now, rather than later.

Within public/images/categories create a directory for each of the top level product categories. Put inside this directory a 120x80.jpg image of all products in that category, and of all of the composite images being used as sub-category images within this category. [ Where are the xxx's now? all in Modalities, or does Modalities contain only the ones that are really for subcategories of Modality. If the later, then batch convert the rest from the composite\_png files that Bill had Mandy send. Eventually go through them and pick out only the ones that are being used as category names, put those in /public/categories/all and adjust the program (views/categories/show.html.erb) to find them there. For now, add the subdirectory name into the pathname being used to find the category images. WARNING: this may not be possible with a category tree in which either the 2nd or 3rd level nodes can be leaf-level. If that proves to be a problem, copy them all in the /public/images/products/all. That was the case, since the category 'names' in the spreadsheet ( and therefore the database ) did not have underscrores, and the filenames did. ( And I didn't want to insert an obscure a regular expression to replace the spaces with underscores. ) However, images/categories/all only has the XXX images for modalities, not for the other top level categories. Where are they? Got them; have inserted the thumbnails that are used in the category landing pages in in /views/categories/alls. [DONE] 12/9

Go back and recreate all of the 120x80 thumbnails, shrinking the 169x179 jpgs (or the original pngs to 80 vertical, and then growing the canvas to 120x80, and adding a white background. Then sve the file to /images/cartegories/<correct category> or just /images/categories/all if don't want to keep a copy in the directories also.

— modalities [DONE] 12/10

— clinical electrotheraphy (6 images: 7100xx-000 where xx=16/25/31/33/54/65) [DONE] 12/10

— [ etc ]

— clinical supplies

— tables and traction

— fitness productds

— orthopedic products

— AT/Taping

— Evaluation

— Dailing LIving Aids

— Lymphedema

— Wound Care

7. insert images for all of the products

within leaf-level category pages

Shrink the roughly rectangular PNGs to 80 high wide and whatever width setting the 'constrain to same height/width ratio' yields. Then place that in the center of a canvas 120x80px in size, and set the canvas background color to white. Export as a jpg image, storing the result in a new directory: /public/images/products/thumbnails.

modalities

clinical electrotherapy

ultrasound [DONE] 10/12

stim

combination

for product\_types and products

Right now they are in /images/products/<top\_category\_name>. They are all 169x179 jpgs converted from the larger pns is /TruMedical/Content/.PNG Images all categories from Mandy Aug4 Sept27. ~706 of them in total across all categories. Normalize to rectangle in golden-mean ratio — or a rounded-to-the closest 10 pixel equivalent. Check GettyImages standard ratios.

Have leaf-level category pages find them within their subdirectories by adding the subcategory name to the image\_filename? means first substituting underscores for spaces in [sbu] category name.

7. Within the partials:

views/home/\_subcategories.html.erb

views/categories/\_leafCategories.html.erb, and

views/leaf\_categories/\_products.html.erb

— allow user to click anywhere on the subcategory block (image OR text) to get to the next level of subcategory or product.

— same for links to product pages from leaf-level category pages: clicking anywhere on the block should take you to the product page, not just on the live text link.

— highlight the text link on hover of the entire block (not just the text link); use the site orange rather than red that is currently in there; No: replace it with turning on the orange background for the entire blocks as doing on level-2 category pages.

8. the partial \_products.html.erb currently assumes that it will find the product images in images/products/modalities: generalize this by point it at images/products/all\_small and moving images on 120x80 (120wide,80 high) white pads into /images/products/thumbnails.

9. The show action in the LeafCategoriesController makes a similar test assumption: it always returns in @products the set of products from category 9. It needs to accept the category number from an argument on the call. Thta argument needs to be set up in the link\_to macros in views/categories/show.html.erb.

10. sort [sub]categories within category so that things are shown on the page in the order specified in the category-tree entries.

11. get left-nav partial to work on category pages as well as home page.

12. remove the sibling nav: just confusing; Instead make the entries in the bread crumb trail live so that a user who wants to see other altneratives can go back up the tree to the [sub]category landing page that got him to where he is now.

13. left nav currently closes the fly-out panel for one menu item when you hover over another. That doens't work for the last one, if you, e.g., move your mouse from payor to either of the blocks in the rectanagle with Tru-Value products; in that case, since you haven't moused over another menu option, the 'payor' overlay remains up. Could we add a special-case mouse leave event to just the last item in the menu list without screwing up the browser's event stack state? If not, how about explicitly flushing the browser event queue? The API is probably unique to each different browser. Alternatively, put the code to close existing menu overlays into the event handlers for mousing over Promotions or Tru-Value Products. Clearer — and need to define what Bill wants to happen when the user does mouse over these elements anyway. One potential gottcha on this tack: Promotions and True-Value Products are bulit in on partial, the menu in another. Would need browser state to be visible across the two. Should be Ok.

**6. product pages**

1. insert a simple product page, so that Bill can walk down the tree and get to the bottom. Make sure we can handle branches of the product tree that are only 2 levels deep as well as those that go down the full 3 levels; this is a navigation issue, not something for the product page.

2. Note: the product images in images/products/all\_small and all\_medium DO NOT include all fo the products. I could not even find the ten images for modalities/electrotherapy/ultasound — unless for the product 7102490-002 and 710249-005 and 710249-010 we always use the image with name 7100249-000.jpg ( which is what the \_product partial does not ). WE ARE MISSING one image for our test subtree: 710032-000.png

3. insert working buttons to add-to-cart and checkout so that we can test the full path to ordering a product or set of products ( before we necessarily have the product group pages working as in the catelog. If we cleaned up the scrollers on the home page, in inserted the search, the sammon-part-number search, and made the catelog request form create [user]<-->>Catelog\_Request records so it would work with existing users rather than always assuming that they are new, we would have a site that Bill could put up, deciding the add the product group pages in a subsequent point release. [ Footnote: we would have to get the matrix item support working so we could handle colors and sizes in any such release 1.0. ]

**7. product set pages ( aka 'matrix' products pages; currently, in the code base, 'base\_products'**

1. reread my notes

2. review the design of the database classes/tables: reinsert the 4th table: line, type, model, configured\_product(\*), rather than trying to collapse it to 2? If insert Rails-side Admin subsystem then modeling the product class hierarchy completely is not much harder to administer than trying to keep it brief enough that we can push it into the Netsuite 'Item' table in an 'encoded' form. The downside of the encoded-in-the-Item table approach is that the site administrator would always have to remember why it was there, and how to work with it. Would also add all four the record types to Netsuite, but if we put the Admin subsystem on the Rails side, then we really wouldn't need anything on the Netsuite side other than the single predefined 'Item' record type with the built-in 'matrix' option enabled for the products that come in color/size combinations — essentially the 'configured products' that are actually being ordered, shipped, and tracked in the Netsuite financials.

3. create them with migrations

4. load them with data

5. write the HTML/CSS/Ruby for the pages/overlays involved.

(\*) How about Line, Product, Model where:

Manufacturer <->>ProductLine

ProductLine<->>Product

Product<->>Model

If a product has models, then it is Models that correspond to Netsuite Items

If it does not, then it is Products that correspond to netsuite Items.

Note that some products may come in sizes, and colors. This is handled by defining 'matrix items' on the Item in netsuite, one for size, one for color, and passing in orders for a particular Product (=Netsuite Item) that comes in a particular size and/or color.

Accessories are defined on Products ( not Models ) — in the case of the Intelec Transport Ultrasound unit, the unit is the Product; it comes in 4 models depending on the size of the applicator, and it has as accessories: cart, battery pack, carry bag, and any of the independently orderable applicators.

The only odd bit about this, is that the extract of items from Netsuite would be loaded into what? the Model table? And lots of Products would have a single model? The other alternatives is to use the word Product for what is here Model, and then introduce Product\_Type for what was here Product. Then netsuite items ( or at least that those don't have matrix parameters) map to Products. And we have to invent a term for the combination of an item in a given size and color. I suppose we could refer to that as a Product in a given size and color, or a Product Configuration. The problem is that English tends to use the simple term ( in this case 'Product') for the thing you think of most immediately, and qualifiers for less common things. And in this product set, when you are thinking about the simpler ones, model#2 fits this bill better. When you are thinking about complex ones, model#1 fits this bill better. I'm going to go with model #2 because it fits 80% of Tru-Medical's product line, and accept the oddness of having Product\_Types in the complex case.

So for the complicated case we have:

Manufacturer <->> ProductLine

ProductLIne <— contains —>>ProductType

ProductType <— has—>> Product

Product <— comes in —>> [ Sizes, Colors ]

and

Category <—>> Category, and we distinguish two subtypes: top-level and leaf-level categories

leaf-level Category <—>> Product\_Type

or?

leaf-level Category <—>> Product

We could hand this by introducing product types even for the products that don't have them ( that is, in the language of model#2, products that don't have models )

Or (option B) we could do it by defining on the Category record, both the associations product\_types and products. Then in the leaf-level category show.html.erb page, if the product\_types association is empty for the category, we list all of the products. Otherwise (if the product\_types assocation IS populated) we list the product\_types. And the product\_type show.html.erb page will have on it all of the products and accessories that go with this product\_type. Or, to try to bridge the two sets of terminology, we might say that a product\_type has ( or 'comes in' ) one or more models, each of which is referred to by the inventory and financial systems as a separate 'product'.

**Ok, so we're going to go with model #1 (ProductType <—>> Product ) and Option B for mapping leaf level categories to ProductTypes and Products.**

Mechanics of implementing it, from where the software stands now:

1. We already have the models: Category, ProductType and Product.

2. Category already has a has\_many :products relationship defined on it; add a has\_many: product\_types relationship. [DONE] 12/10

3. ProductType already has a has\_many :products relationship defined on it. Add a belongs\_to: category relationship. [DONE] 12/10

4. Product has a belongs\_to: product\_type. Add a belongs\_to: category. [DONE] 12/10

5. For the complex case, add classes Manufacturer and ProductLine. Each has a name, and a logomark. ProductLine has a description also. link ProductLine <->> ProductType. [DONE] 12/10

6. Create/apply a migration to add foreign key for product\_type ( in addition to the existing one for leaf-level category) to products. Add column for this to the excel sheet from which the products table is being loaded; and reload the database [ populate this column and reload the database again when get to building/testing the complex case below ] [DONE] 12/10 used products\_all\_wModels\_1018.xls as base.

7. Modify views/leaf\_categories/show.html to see whether it should be listing product\_types or products, and call the appropriate partial. How? For now, put a column has\_product\_types? in the Category table. leaf\_category/show.html.erb line 216ff. [ Test the if ] [ in Modalities/Electrotheraply, chose Accessories or Lotions and Gets. Nether has the type/model distinction and should therefore render the partial 'products'. Does it?

8. A partial for Products exists. Create a new partial for Product Types. [DONE] 12/10

9. There is a views/product\_types/show.html page. Check it when get to step 13.

10. Build a views/products/show.html.erb page [DONE] 12/10 —

— get it to compile; [DONE]

— get it to work: may need step 12 before can test it; need a path of pictures through the tree of landing pages to get here.

problem: the category\_id' foreign key in the file used to load the products table is not filled in, so we don't know have the llc<-—> products mapping we need.

solution: Build a ruby script that can be run in the Rails console to fill this field in in the database records after the products spreadsheet is loaded, to wit:

Product.all.each do |p|

p.category\_id = Category.where( :name => p.category\_l3).first

p.save

end

To avoid problems with l3 category names that occur multiple times in the categories table (e.g., 'Accessories' ) will have to do this more rigorously eventually, but this will serve for the testing of the product and product\_type pages I am trying to do now. It will return the correct Accessories for the initial product subcategory: Modalities/Clinical Electrotherapy.

Returning nil. Why?

Strip products file down to only the products in top level category Modalities to make it easier to look at the loaded file using Razor or Rails console

Reload products table from this file. OK

Confirm that it has 192 products by looking at it through Razor. OK

Try rerunning Ruby script through Rails console against this smaller table again: Runs, but sets all category ids to 1. Why? That would be the case if I were matching against p.category\_l1, but I'm not, I'm matching against p.category\_l3. Look at excel file used to load categories table. Look at categories table.

solution: original Ruby macro confused the foreign\_key (category\_id) and the instance of the class Category that this id pointed to. Macro should have been:

Product.all.each do |p|

p.category = Category.where(:name=>p.category\_l3).first

p.save

end

11. Create a products controller: products\_controller.rb, and put a show action in it. [DONE] 12/10

12. Put in the pictures to get us down to an example of a simple product ( no models, no sizes, no colors).

— pick one: [DONE] 12/12: Modalities/Clinical Electrotherapy/Lotions & Gels

need 4 additional pictures: 710063-00, 710064-000 710066-000 and 710072-000 in the directory /public/images/products/all\_small.

problem: There were no PNG images for 710063-00 and 710072-000 in the directory Bill resent through yousendit last week. Resized the others (..64 and ...66) and placed them on 120x80 white pads in images\_small and 169x179 white pads in images\_medium.

— test the products page built in step 7 above: confirm that you can

(i) get to it from leaf-level category page, and

problem: 1st category in sibling nav is always highlighting, but not set to Lotions & Gels. [ forced for test; rebuild or remove sibling nav ]

problem: one product title is running to a 3rd line and therefore falling outside of the product block in the leaf-level category page. Limit to <n> characters, or change entry in database.

problem: @product instance variable getting passed to Product#show.html.erb is nil. Check controller; check argument being passed through link\_to macro in \_product.htm.erb partial. ( and once confirm that that code works, move it back into partial; it has now been pulled forward in to the body of leaf\_category/show.html.erb ). Was calling ProductTypes#show not Products#show. Products#show was not setting up the instance variables @product and @category needed by products/show.html.erb.

problem: smashed page layout: nothing in main section; footer floating up into main section. Solution: removed divs involved in tabbed interface triggered by sibling nav. Old code kept in show\_wMavics\_Tabs.html.erb.

OK: better: page layout no longer broken, table of products appearing, but individual product blocks are stepping down and to the right rather than lining up; 'a product' looks like its outside the block, etc [FIXED] 12/12

problem: views/products/show.html.erb breaking: not showing product image/description. [FIXED] see below

Removed the tabs at the bottom, cleaned up div structure. Reinsert them using jquery-ui tabbed interface? Mavics one stored in views/products/show\_wMavicsTabs.html.erb. Still no content showing up. Why? answer: unclosed <script> tag.

problem: image not in #left column, description in #right. [FIXED] 1212: redefine them locally within the heading of this file

problem: (TM) not being interpreted; showing up as '&trade'. [FIXED] 12/12

(ii) push orders from it into the standard Netsuite shopping cart

(iii) create a more subtle acknowledgement that you have put an order in the cart

(iv) checkout and confirm that the order flows correctly through to the Netsuite backend. This means that we will need skins for netsuite generated pages. What should they look like?

(v) UI/layout changes:

— large image pictures in product pages?

— zoom on product pages

— use 169x179 images in leaf\_category pages?

— reinsert set of tabs in the bottom of products page. Eventually where will want to put clinicial\_comments, users\_comments, videos on how to use, msds, ... . Ideally would make the presence of individual tabs conditional on whether content for that tab existsfor this product, and the whole tabbed block conditional on whether some tab has content for this product.

— large background images per the Mavics site — but healthcare focused. Pick a couple samples from GettyImages and show Bill what they will look like in the site.

13. Put in the pictures to allow a site user to get to an example of a product that comes in sizes and/or colors:

— pick one: Modalities/Accessories/Nylatex Wraps [DONE]

— Revise the database model to carry model product sets explicitly [DONE] 12/29

introduce class ProductSet with 1:m relationship between

ProductSet <—>> Product [DONE ]— I named it 'BaseProduct' though rather than 'ProductSet' [ Consider going back and using BaseProduct for ProductType, and ProductSet as initially specified here: for the matrix products. In a clothing company you might think of these as the same product that comes in differen sizes; here they really are more like product sets: if the cold wraps come in different sizes, they look very different from one another, and are used for different parts of the body, i.e., they are a set of products, not a single product ( e.g., a T-Shirt) that comes in different sizes and colors. This would eliminate the awkward term 'Product Type' . ]

Modify define of Product to add foreign key: base\_product\_id, and a boolean field to say, for each product, whether it participates in a product set. [DONE] 12/29

create a separate table for product sets — copying information from the products with extension 000 that also have 001, 002, ... 00Ns. [DONE] 12/29

Populate it from the Products table using a Ruby script run in the Rails console

use a Ruby script to set the foreign keys in each of the product records that participate in a product set to point to the product set tuple using a Rails 'id'.

use Rails console to test that for the product set Nylatex Wraps, we can retrieve all of the individual products in that set. [ Do we want to? Or do people just buy from the list as in the catalog? Since there is no additiona information associated with the individual products, I think we leave it that they buy from the ProductSet page. ]

also test that what shows up in the leaf-category for Modalities/Accessories is only the product set Nylatex Wraps and not the individual products within that set. This will mean modifying the view/leaf\_categories/show.html.erb file

Built a views/product\_set/show.hmtl.erb page that displays the product set: image, description, and the list of tms-part-no, size [ and color? ] , price for each of the products in the product set. [DONE] 12/29.

Issues this raised:

1. Add a column to the left of the tms-part-no for letters used to tie the individual line item to one of the images in the collage picture if it is one

2. ProductSets with long names are breaking the sibling nav

3. Images in ProductSet (aka BaseProduct) pages need to be unsquished.. Seems like this could be done as a batch process from Mandy's png originals.

DONE 12/31 for images in the Modalities top level category that we are using to test the three types of product page: simple, matrix, complex

Do it for each of the other top level categories

4. Zoom? on ProductSet image as well as on individual Product images?

5. Some ProductSet descriptions (eg., Nylatex: asterisking which productds are in some group ... ) need to be edited to remove stuff that is done in the catalog but not on the web site

6. Reduce vertical spacing between lines in the products table. ( Tried without much luck 12/31. )

7. Formatting: width of the @products.description field in products that have either short descriptions or ones for whom a wider field works better. Made them all 300px: put <br> tags in descriptions that aren't going to fit in this. Or make them 100% width and insert variable width line of dots before the price.

— test the path from category to leaf-category that contains one or more product sets

— test path from leaf-category image for a product set to the product set page

— test that you can push a suitably qualfied order from the product set page to Netsuite shopping cart ( Might eventually, on Rails-side cart, allow peopel to change the color of something already in the cart, but with NS it will require delting the old items nd inserting a new one. )

**8. complex product pages ( currently 'product\_types' )**

1. hand build the database for the electrotherapy example of the complex case. See that you can get to the product\_type page, and order products of this type, accessories, etc direcltly from that page (without going to a 'product' page).

1. DONE 2/31: but with accessories just as products that are references from complex product types through an 'accessories' association.

2. Make the product\_type <<->> accessories relation an M:N one. And if that creates problems with the 2nd association (product-type.accessories), then pull accessories out into their own table (with exactly the same fields as the 'products' table). Actually do this first before trying to make it work with two associations defined on the same two tuple types. [DONE] 1/2

2. Code the #show page for product-types: views/product\_types/show,

1. Code image, product description, accessories, and ancillary information. DONE 12/30.

2. Cleanup:

1. Swap out jquery-ui tabs library for jquery tools tab library so it can be styled it to match the rest of the site. [DONE] 1/1/2012

2. Remove declaration for the association product\_types.accessories that goes directly to products from product\_types.rb. See if this allows p.save when adding timestamps to all records in the 'products' table to work. [DONE] 1/2. Yes script to add timestamps works again.

3. Insert new 120x80 category images when Mandy sends them. [DONE] 1/2

4. Pull PK's code for add-to-cart, checkout from heroku and insert into latest page templates for simple, matrix and complex product types. [DONE] 1/2

— addToCart button on views/product\_type/show.html.erb not working

5. Pull PK's code for register, login, show\_cart, checkout for the upper right and insert into views/layouts/application.html.erb. [DONE] 1/2

/////////////////////// as of 1/2/2012 //////////////////////////////////////////////////////////////////////////////////////////////////////////

6. Go through each of the complex product-types within Modalities and see that the correct products and accessories are being displayed. Check for missing images.

7. Consider renaming: base\_product —> product\_set ( so that we can later replace product-type with base\_product ).

8. Remove unused columns from products and accessories tables, e.g., product\_line\_description, and foreign keys used in 1:n product\_types.accessories that preceded m:n version of this association.

8. Check to see where there is information in the catalog about product-types, product-lines, or manufacturers that is not in the database and so is not being displayed on the web site. Should we add it?

**9. Get left-nav to take us down to these three types of product page**

1. make leaf-level category controller strong enough to determining, for each image shown on the page, whether it represents a simple, matrix, or complex product, and set up the links correctly.

2. Are there cases where we have only a 2-level category tree? If so, remove the assumption from the code that the leaf-level category always occurs at level 3.

3. go back and remove all of the testing code in leaf\_categor#show, Products and ProductTypes controllers and views/products/show.html.erb and views/product\_types/show.html.erb. Test that navigation works properly when it has to deal with 3-level hierarchy with these 3 different types of product page

4. Make left-nav work on pages other than the home page. Turn it into a partial that can be included in each of the templates. This means that the popups for each of the top-level categories that are in the left nav have to be available on all pages that show the left-nav. Another partial. [ or else put the top of the left nav ( top level product categories) into application.html.erb along with the overlays that these top-level categories call up.

5. Clean up highlighting in 2nd 3rd level categories so that it matches that in 1st level categories — ie the background of the visible portion of the block ( the portion containing the words below the image) turns orange when you mouse over the picture or the text, and a single click anywhere within the block takes you to the next level down, or to a particular simple, matrix, or complex product page.

**10. Other Stuff**

The following tasks were added ( or made explicit) by TA on 1/1/2012:

1. Rebuild breadcrumb trail to walk up category tree to root regardless of how deep it is. Make it into a shared partial, and rewrite all views that include a breakcrumb to render it as a partial.

2. Clean up the UI associatedwith Google CSE search button.

1. leave it where it is and do something like Mavics does — essentially unwrapping it when someon clicks on the button sitting in front of it, and at that point exposing the text entry field, the search button, and the clear-text field icon ('x'). Consider doing that in a popup window if can't fit it into the slot in the left-nav reserved for it.

2. alternatively, move search, register/login and view-cart / checkout to the upper right, above the floating white panels on the screen

3. Insert search by competitor's part number. Assume there will be multiple columns within the spreadsheet used to load the database, for at least 4 competitors. When user types a number or string in the search box in the page footer, search for matching numbers in any of the columns of competitor part nos within the product table. Bring up a popup window to say the search is in progress, and then either a 'no matching part no's found; please call us on our toll-free line at 800-999-9999' or a line that pairs the # entered, [ the compeittor name], and the tms-part\_no, and says whether the match is exact or only similiar. Give user a button he can click to go to the product page for the tma product shown as a match. = top level item #3.

4 Get scroller on the home page to work, with text blocks appropriately positioned on each of the images used.

5. Get sibling nav to work properly or eliminate it

6. Layout: line up top of the #right-content thing, when it is on the right, with top of middle column.

7. Do a R2 sample of what a complex type would look like if had multiple images and zoom.

8. Put in Admin subsystem for controlling the content in the Rails-side DBMS. [ Discuss the Rails -> Netsuite autosynch as an R2 addition with Bill.

1. Three options for synchronizing product data: (i) general schema mapper built around open-source T\_\_; (ii) schema mapper for TruMedical database and Netsuite setup; (iii) break data on the Rails side out into the same tables we are using to support the object model in the Rails site.

2. Pushing images to Amazon S3 and synchronizing.

3. [ Pushing Video to a CDN: current HDDN, Akami/Brightcove, [ ... ]

9. Deal with different price levels being displayed for different categories of user, and users from different clinics which have negotiated different price levels.

1. Does this imply a register/login subsystem on the Rails front-end so that we know which prices to show the user, and

2. A way of passing an argument on which price-list to use to the Netsuite back-end either when the user logs in, or when invoke the view cart and checkout operations.

10. Cleanup the code base, consolidating the css libraries and removing any css libraries that are not being used, and any javascript libraries that are not being used.

11. Home page overlays that run off the bottom of the page: scroll bars within them? realign company pages so they go from the bottom of the left menu item up, instead of from the top of the left menu item down?

— waiting on Bill for content for 'providers' section

— get some images into the Tru-Medical, and Patients/Providers/Payors overlays? They are a bit text-heavy at this point.

12. Promotions and Tru-Value product blocks in the #right-content section: What does Bill want us to do when someone clicks on one of these:

1. Promotions: [ ... ]

2. Tru-Value Products: [ ... ]

13. Prices show on the site vary by who is logged in. Currently just using clinical list price: good enough for Release 1? If not, this seems to require moving the login/registration subsystem to the Rails front-end.