**Multiple Price Schedules**

**The real-world situation**

There are two kinds of users: consumers and clinicians

consumers get one on-line price: price\_consumer\_online, and may not be able to order some products that are 'clinician only'. The 'consumer-online price' is a new one, currently the same as the Netsuite price 'clinician online'.

clinicians come in two varieties

— those that are not affiliated with the clinic with whom Tru-Medical has an contract; they see 'price-clinic\_online'

— those that are affiliated with a clinic with whom TruMedical has a contract. They see whatever price schedule their clinic has negotiated with TruMedical.

There are some cases in which a particular clinic has negotiated additional discounts for some specific products from the base price schedule they have been assigned.

We need to see that once, a clinicial has registered ( so we know what clinic he is with), and logged in, (i) that he sees prices for all products that clinicians can order on line, (ii) that he sees the correct prices, and (iii) that when he puts an item in the cart, it shows up in the cart with that price . (ii) and (iii) rather than having him see one set of prices on the web and in his shopping cart, and knowing that at checkout these will be reduced to the prices in the schedule negotatied by his clinic with Tru-Medical.

**Options for modeling this in the software**

option #1. Registration/price levels happen on the Netsuite side:

to wit: The register and login links on the pages of the site go to the standard Netsuite routines for adding something to a cart, viewing the cart, and checking out. Custom Netsuite records are created to support 2 different types of site visitor: consumers and clinicians. And clinicians are linked to a new custom record type added to Netsuite: CLINIC. The registration screen is amended to ask for his clinic. When he logs in, netsite calls back to Rails passing it an index to the price strucdture to use with this client. Means (i) setting up a listener on the Rails side, and (ii) modifying the Netsuite login function to send an HTTP msg to the Rails listener that includes the user name and the price schedule identifier. The Rails listener updates the Rails database to note the price schedule associated with the current user. Any Rails-side code that lists products (ie., views/products/show.html.erb, ... ) looks up the price schedule applicable to their current on-line user, and applies the prices listed therein.

option #2. Registration/login/price-levels happens on the Rails site

to wit:

1. modify data model

clinician subtype of user

consumer subtype of user

add clinic, include price schedule granted them

set up n:1 relationship between clinicians and their clinic; modify model/ ... .rb files

2. load the data for existing users, clinics and the price schedule they have negotiated.

3. get some sample clinics and clinician names from Denver.

4. Insert them into the tm202 database for testing.

5. take out Mavic-like orange blocks that control which of the images in the image carousel we are on now.

6. insert Rails-side registration/login system

7. modify it to capture information on the clinic, and to link clinicians to their clinic.

8. modify views/products/show to use this information to determined which price schedule to use.

9. add these prices to the HTML message sent to Rails for add-to-cart. [ PK says this is a standard Netsuite-supported option; What is the additional argument that needs to be passed? Modify the javascript in the site to pass it.

10. Make sure the correct prices show up in subsequent call to 'view the cart', and again in the Rails 'checkout' sequence. Both of these are still handled on Netsuite's secure server.

We are going to implement option#2.