Subject: Re: Development Diary (markcarranza)
Posted by markcarranza on Thu, 20 Dec 2012 03:53:01 GMT
View Forum Message <> Reply to Message

Thank you for your thoughtful and detailed reply.

Please remember the list was in language specifically for this client, but I will expand as best I can.

1)
>user interface not good for client services
I'll get to this, first though,

>more for programmer types
forgive me, this was terminology meant merely for the client's ease of understanding.

>user interface not good for client services
My user community, 'client services,' are very, very unskilled with computers. Most unfortunately, the client requests the new application look as much like the legacy app as possible.
>not good
The Radicore UI simply doesn't work the same way, and 'not good' for their purposes. It would be too tedious to go into all the differences, but for this client I need to radically change the presentation layer.

That said, I initially found many aspects of the Radicore UI unfamiliar and confusing. There are things I like and things I don't like, UI idioms that frankly disturb me. I'm hoping to have the time to expand on both, but was planning to to so over an extended period of time. If I'm lucky enough to be able to use Radicore in this project I intend to do so.

I'm an application developer--analyst, designer, and coder--but I mostly focus on 'cognitive ergonomics.' I find out how prospective users think about their information, and design the application to fit their expectations as much as possible, so that it's intuitive to them. This often means spending the predominant coding time on a highly customized UI. But as the end users typically use the app most of their day, any ease, comfort, predictability, and even beauty I can create for them, I feel morally obliged to do so. Often in the past fighting ideas of managers in favor of these values for end users. I'm highly opinionated in this area, but alas, so busy I'm not inclined to write about software usability unless paid to do so. I fully appreciate all the writing Tony Marston and others do, sharing their wisdom, and even more when they engage in discussion. Kudos! Or as kids today say, Props!

>idioms that frankly disturb me
Please forgive me if I only have time for one example at the moment.
If I'm naming this correctly: Task buttons
An example is on page "List Task (Process)"
- There are (I think) 14 buttons besides standard read, update, delete
- way too busy visually, hard to read, harder to find correct button
- buttons should be grouped visually by category of task
- button locations should not change when screen re-sizes

- user-setting-optional tool tips should concisely describe function of each button
- Number terms used across Radicore like (1)(2)(3) in 'Navigation(1)/Navigation(2)/...', or in 'List1/List2/...' should be replaced by semantic terms or abbreviations
- buttons always present like standard: read, update, delete should be in their own standard area, visually separated from custom options, as close to data as possible
I hope this helps explain somewhat. I do understand and appreciate the ease of creating these buttons in the menu/task system, but to me the appropriate tradeoff has to be for end-user ease of use.

2)
>likely need to edit core files, Radicore upgrades would need programmer interaction
Client has no dedicated in-house developers and is sensitive to maintenance issues, from previous coders and Zen Cart upgrades. Because of significant customization, sometimes done by less-experienced coders not taking time to understand ways to customize without making changes to core code, base-system upgrades require 1) using difference checker like WinMerge, and 2) someone familiar with system and past changes.

>core files
With Radicore I might, for example, need to change user_IDs from VARCHAR to INT. I do not know, but prudence tells me to anticipate need to significantly change many core Radicore files.
>programmer interaction
Upgrading to a new Radicore version, under this scenario, would require significant effort and an above-average PHP coder, i.e., someone not in-house. Again, the language and sensitivity concerns this particular client to help them understand issues to make a decision they understand.

3)
>special security needs requires audit of source and upgrade code
This is a medical-services client, the new software will contain highly confidential patient information. The FDA treats such software as a "medical device" and has appropriate regulations as to the history and source. By analogy, in a patient's test, each chemical used needs to be recorded by source and batch number to be tracked back to the supplier, who can further trace it back to the maker. Maintaining a record to each developer (and their resume,) to touch the software gets tricky with open source, but it's an example of the special situation.

>the same action in other frameworks
The real question for this client, under these circumstances, is to use any outside framework at all, or one not from a big company (read here: liability). The client needs to able to show FDA auditors 'due diligence' protecting confidential information: dotting i's, etc.

I'm specifically recommending Radicore for the architectural clarity, but relevantly here
1) built-in audit: all patient information has to have 'provenance,' who entered, when, what changed in record, when.
2) workflow: client doesn't have a workflow-aware computer system and desperately needs one given all the regulatory flows

4)
>possible intimidating learning curve for future maintenance PHP developers
Again, the real question for this client, with no in-house developers, is to use any framework at all.

The CTO strongly prefers 'plain vanilla' PHP, i.e., minimal third-party software, and then only things like jQuery and the simplest of plug-ins; and not say, client-side frameworks built on top of jQuery. The client's questions are: how many developers know Radicore? How easy will it be to find a (when I'm gone) programmer to maintain the system?

They are used to under-performing, difficult coders. In San Francisco in particular there's a currently a tremendous shortage of any good developers, because of the startup culture here.

5)
>single person company/support
This, and the 4) above is much more a bus number issue.

6)
>no unit tests, test suite, automated tests
I'm a tricky person, I put that in there to provoke conversation with the client about testing.

>The cost of producing unit tests substantially outweighs the benefits
I tend to agree, I've never written a unit test in my life. I've primarily been an enterprise developer.

That said, San Francisco is a Ruby (and/or) JavaScript (and/or) mobile (and/or) fondle-slab culture, all pair-programmingy and TDDey. If I had funding (someone else's money) and was quickly developing a commercial-quality mobile app, I'd have (probably young and less experienced) employees do pair-programming and TDD. With the right money, time-criticality, audience-size-based-support-issue criticality, and less-experienced programmers, I sense it's a pretty good idea.

7)
>no needed HTML reporting, only PDF
Imagine changing all existing Radicore PDFs to web pages, with (optional) print button and print css. I don't know Radicore enough to know how easy or difficult that would be.

The client needs many PDFs but many, many dashboard-type web page reports. Again I don't know Radicore enough... easy or difficult... But haven't seen in base Radicore. Haven't installed example systems yet.

I like the way Radicore architects PDF reports.

>What sort of reporting capabilities would you like to see?
Hoo boy, I'd love to see a system allowing user-created queries and reports. Short of that, I'll have to get back to your question until when the issue is in front of me.

8)
>The UI is tremendously too busy
I'll stand by that. I hope I've helped explain 'too busy' in the example above.

>when you implement your own application using Radicore
>to change the way the standard screens look

I understand. Haven't installed example systems yet, it would be great if one showed significantly different look and feel.

>not the simplest possible for the user community I have to support
users = low computer skilled, typists

I'm going to have to really think through a clear, simple, and predictable UI for this particular client.

A few examples of rich-client UI control surfaces the client is used to, commonly available in Windows-based enterprise apps are

- top-row dropdown menu
  - visual information hiding
  - discoverable with mouseover
  - calm (always the same, never changes from screen to screen)

- left panel context menu
  - sometimes in task window
  - choices blank or grey-ed out if editing

Now, some of these are not common web page features. Certainly web pages are commonly built more for mouse interaction than for keyboard interaction.

But the kind of app the client has, and if at all possible the kind of app I intend to build, needs to be easy to use without a mouse.
The Radicore UI features I've seen so far are simply not optimized (completely, there are some features) for high-volume data entry in this way. This is not easy with web-based interfaces but requires JavaScript.

>If you want to use JavaScript
I understand

>separate screen based on the SEARCH pattern
>Quick Search bar
I merely suggest the different functions be differently named. 'search' and 'quick search' is sufficient, but not necessary. Or 'search screen' and 'search'. IMO, two buttons with the same name makes the user think unnecessarily. It's a relatively small point, but important to me.

>Quick Search bar
I was confused when I entered text and hit 'search' and nothing happened. As a user, I felt stupid. "What did I do wrong?" Making any user feel stupid is morally wrong, completely common, and typically preventable.

>if 10 or less records, hide paging top (show...) and bottom
The paging controls are useful but not visually pleasing. But they are only useful when they're usable. Many design principles follow the pattern of Strunk & White's dictum "omit needless words."

>Why?
Unusable controls are 'behavior-free decoration,' like Edward Tufte's chartjunk (content-free decoration in a visual display of information)
UI controls that are unusable just add visual clutter and confusion to an interface.

The larger value is expressed in Weiser and Brown's Calm Technology paper
Quote:If computers are everywhere they better stay out of the way, and that means designing them so that the people being shared by the computers remain serene and in control. Calmness is a new challenge that Ubiquitous Computing brings to computing. When computers are used behind closed doors by experts, calmness is relevant to only a few. Computers for personal use have focused on the excitement of interaction. But when computers are all around, so that we want to compute while doing something else and have more time to be more fully human, we must radically rethink the goals, context and technology of the computer and all the other technology crowding into our lives. Calmness is a fundamental challenge for all technological design of the next fifty years. The rest of this paper opens a dialogue about the design of calm technology.

>Nobody else has suggested such a change, so it must be very unimportant.
This is not a worthy statement, and I'll let it pass otherwise unremarked.

>List column heading 'Select' could be link/toggle select all/select none.
These are journal notes to myself that I'm willing to share to give ideas and to get comment. I'm looking for ways to make things simpler.

>If all selected, then 'None' otherwise 'All'.
If there are 0-n entries manually selected, the control remains 'All'. Only if all entries selected does 'None' show.
The logic could also be the reverse:
If there are 1-n entries manually selected, the control remains 'None'. Only if no entries selected does 'All' show.

Just as easily:
if 0 entries selected, show All
if all entries selected, show None
otherwise show All None

>'locked' check box
Thanks for pointer. My default expectation is 'locked' behavior. My tradeoff choice would be to fix global behavior to 'locked' and get rid of the control (on every list screen), but I would check with client and users.

I hope that clears up the most important questions. All I have time for. Well more than, but thanks again and more later, I hope.