Subject: Development Diary (markcarranza)
Posted by markcarranza on Tue, 18 Dec 2012 10:08:53 GMT
View Forum Message <> Reply to Message

Hi, I'm a framework, RAD guy.

UI2 with DBaseIII in 85, Clipper in 87.
TRO (Tom Rettig's Office) in FoxPro in the 90's
MM.NET in .Net C#
Ruby on Rails
I've written a few of my own, including a lightweight PHP framework.

I'm likely to get deep into Radicore for a few months, and to give to and get from the Radicore community, I'd like to record the overall list of suggestions, changes, tips, tricks, pitfalls, successes and get appropriate feedback.

I'm redoing an entire legacy custom MIS intranet/extranet system. I'm recommending Radicore to my client in a way they understand:
[pluses]
+ since 2006, 4 version releases in 2012, well supported
+ highly opinionated, lots of writing, more and better documentation than any other (I know of)
+ free, open source
+ good working code
+ workflow, auditing, data dictionary
+ 3-tier architecture: data access, business logic, presentation logic
[minuses]
- user interface not good for client services, more for programmer types
- likely need to edit core files, Radicore upgrades would need programmer interaction
- special security needs requires audit of source and upgrade code
- possible intimidating learning curve for future maintenance PHP developers
- single person company/support
- no unit tests, test suite, automated tests
- no needed HTML reporting, only PDF (CTO: user network user temp area traffic/space issue)

The special situations of this client are:
- (very) low computer-skill employees
- high security
I appreciate all the work that has going into the UI standards of Radicore:
- classes of controls always in the same places
- handling of multiple records, and Parent records in hierarchy
- paging
But the UI is tremendously too busy, not the simplest possible for the user community I have to support. I have to radically minimize the choices, degrees of freedom for the user. Make it perfectly clear, with the minimum of choices on screen at any one time.

For example, the Radicore UI has the idiom:
1) display all tasks for context
2) select record (or records) from list

3) choose task
Where I need to have an idiom:
1) select one record from list (click)
2) display screen of (most important to see, dashboard) record data
3) display tasks applicable to record (edit, delete, ...)
4) choose task

I'm not sure if this above explanation is clear enough, but I hope future application screen shots will help explain. It will take rethinking of UI side of transaction patterns, if I use them.

For employee-facing screens, I'd prefer using a rich-client JavaScript MVC framework using AJAX and JSON data from the Business Logic layer, but that will be decided later.

The first step is a data element inventory of the legacy system. I've created several Radicore Subsystems matching major legacy components and imported those legacy tables and columns into the Radicore Data Dictionary.

The Description field holds the new column name, existing data examples and problems and a '(?)' where there are discussions needed.

The Comments field holds the extended data element description and sample data, business rules, presentation rules, valid ranges, etc.

The Required field is used, but most other Data Dictionary fields are unused because this is a documentation task preceding database redesign. The 'legacy' Data Dictionary will never be used for an application. I may add fields (new_column_name, show_historical_change_alert) for new application metadata. I'll certainly create new reports and upload them here.

The next steps before the database redesign are:
- Report Inventory (legacy paper reports in binder)
- Forms Inventory (paper forms (not in software) in binder)
- Screen Inventory (legacy screen shots in binder)
These will be much scribbled on and probably written up in .doc files, notes printed up and included with the examples.

Suggestions, observations:
List Subsystem screen
- two buttons named 'search'
- screen/example search should be named 'Find' or 'Query'
- in column search, when column dropdown is blank, search fails and typed-in search term unexpectedly disappears. Blank column dropdown is useless and broken. Show no blank: first or best column in dropdown. Or change blank column dropdown search to query all text fields.
- if 10 or less records, hide paging top (show...) and bottom
- list column heading 'Select' could be link/toggle select all/select None. If all selected, then 'None' otherwise 'All'. JavaScript required.
- not sure what 'locked' check box does, not discoverable or documented in http://www.tonymarston.net/php-mysql/menuguide/mnu_subsystem(list1).html
- list column heading 'Count' clearer as 'Tasks'

Thanks! --Mark

Subject: Re: Development Diary (markcarranza)
Posted by jjtoranzo2004 on Tue, 18 Dec 2012 13:56:11 GMT
View Forum Message <> Reply to Message

Mark,

The behaviour of searching all searchable fields (in "column" search) is exactly what I want. I am sure that there is enough information in the data dictionary to make that behaviour not so difficult to implement. The data dictionary knows the type of every field and also knows if a field is searchable or not.
I know we can always customise a task, but the desired behaviour is general enough to try to incorporate it to the framework. Perhaps we can convince Tony.

Thanks for sharing,
Juan

Subject: Re: Development Diary (markcarranza)
Posted by AJM on Wed, 19 Dec 2012 21:41:54 GMT
View Forum Message <> Reply to Message

Searching all searchable fields from a single input field is not an option which I have seen, or ever been asked to provide. Most people know in advance which field they need to search in, so searching all possible fields would seem to be an unnecessary overhead, especially when one of the fields could be in a joined table. Have you run benchmarks to see how it would affect performance? This would not be an easy option to implement, certainly not as easy as you think.

Subject: Re: Development Diary (markcarranza)
Posted by AJM on Wed, 19 Dec 2012 21:52:55 GMT
View Forum Message <> Reply to Message

While I welcome comments and criticisms, I feel that some of your points do not have enough substance for me to work with. Let me try to address your negative points one by one:

1) user interface not good for client services, more for programmer types

Can you define "not good"? What changes would you make? If it is fonts, colours and positioning that you don't like then you can easily add your own CSS files, as explained in The use of Cascading Style Sheets within Radicore

Radicore was designed to aid in the development of back end administrative applications which

are used by members of staff, and not front end websites which are used by casual visitors. Having functional screens which give fast access to the organisation's data has a higher priority than sexy screens full of gimmicks and fancy images. I have written applications with Radicore which are regularly used by non-programmers, and they do not have any difficulty using the screens.

2) likely need to edit core files, Radicore upgrades would need programmer interaction

Radicore is not meant to be used by non-programmers. It is not an end-user application in its own right; it is a tool which allows programmers to develop their own applications more rapidly, with less effort, and with more features than other frameworks.

3) special security needs requires audit of source and upgrade code

What special security needs? Would these needs require the same action in other frameworks?

4) possible intimidating learning curve for future maintenance PHP developers

There will be some sort of learning curve with whatever framework you choose. You should also consider the fact that the quicker that something is to learn then the fewer facilities it has to offer. Radicore is a bit like PHP itself easy to learn for simple tasks by providing default behaviour which covers all the basics, yet flexible enough to allow virtually all defaults to be overridden or replaced with you own code which you can customise to your heart's content.

5) single person company/support

Some people would see this is as a benefit as you don't have a group of different people with different opinions trying to push the product in different directions. Have you ever heard the saying "A camel is a horse designed by a committee".

6) no unit tests, test suite, automated tests

The cost of producing unit tests substantially outweighs the benefits. I'd rather spend my time producing software that works than writing a second piece of software to prove that the first piece of software works.

7) no needed HTML reporting, only PDF

I don't understand this point. An HTML report is just an HTML document, just as a web page is an HTML document. I have regularly produced "reports" which are actually web pages instead of PDF documents, and I usually include the ability to export all the data to a CSV file so that it can be manipulated in a spreadsheet. What sort of reporting capabilities would you like to see?

8. The UI is tremendously too busy, not the simplest possible for the user community I have to support. I have to radically minimize the choices, degrees of freedom for the user. Make it perfectly clear, with the minimum of choices on screen at any one time.

Define "too busy". Every component in the screen has useful functionality, so what functionality

would you remove? You also have to bear in mind that when you implement your own application using Radicore that it has whatever menu buttons and navigation buttons that *YOU* decide to implement, and each screen contains whatever data you choose in whatever place you choose (within reason) using whatever HTML control you choose. The subsystems which are built into the Radicore framework menu, data dictionary, audit and workflow work the way they work and look the way they look, but any additional subsystem which *YOU* build can be customised by you to look and behave the way *YOU* want.

If you really want to change the way the standard screens look you can always try replacing the default CSS files with your own customised versions.

If you want to use JavaScript then go ahead. I do not use any in the Radicore framework itself, but I have built in the ability for you to add in whatever JavaScript you choose. Take a look at RADICORE for PHP - Inserting optional JavaScript

Comments on the "List Subsystem" screen:
- There are two search buttons

One activates a separate screen based on the SEARCH pattern (see http://www.tonymarston.net/php-mysql/dialog-types.html#searc h) while the other is part of the Quick Search bar (see http://www.tonymarston.net/php-mysql/dialog-types.html#title -bar) While they both allow search criteria to be entered they do it in different ways, one quick, one slow.

- search should be named 'Find' or 'Query'

I have known this type of screen as a "search" screen for several decades, and I see no point in changing it.

- In column search, when column dropdown is blank, search fails and typed-in search term unexpectedly disappears. Blank column dropdown is useless and broken. Show no blank: first or best column in dropdown. Or change blank column dropdown search to query all text fields.

By "column search" I assume you mean the Quick Search bar, which works exactly as documented and does not need to be changed.

- if 10 or less records, hide paging top (show...) and bottom

Why? Nobody else has suggested such a change, so it must be very unimportant.

- List column heading 'Select' could be link/toggle select all/select none. If all selected, then 'None' otherwise 'All'. JavaScript required.

To say that this should act as a toggle between select all and select none implies that the selection is limited to either all or none, which is incorrect. It is possible to select any number of entries manually, so what would the next choice be? It could be one or the other, so which would I choose? There is room on the screen for both options, so I will keep both.

- not sure what 'locked' check box does

Take a look in http://www.tonymarston.net/php-mysql/dialog-types.html#navig ation-bar

- list column heading 'Count' clearer as 'Tasks'

OK. I'll change that to be consistent with other screens in the system.

Subject: Re: Development Diary (markcarranza)
Posted by markcarranza on Wed, 19 Dec 2012 22:32:19 GMT
View Forum Message <> Reply to Message

Juan,
>blank column dropdown search to query all text fields
Perhaps, with hindsight, that might be 'text fields in the senior table marked searchable in the data dictionary.'

But, when doing this kind of thing before, the default search choice in the column choice dropdown was 'All Searchable Fields'. In addition, all the column names were in 'English,'; which is a requirements-language shortcut term for 'in the vocabulary of the user, and not the column names in the database.'

An better 'task customization' example would be the Zen Cart customer list screen, where the I changed default search to query if text entered
"WHERE customer_first_name LIKE '%".$input."%'"
. " OR customer_last_name LIKE '%".$input."%'"
if all numerals entered
"WHERE customer_id = '".$input."'"
. " OR customer_zipcode = '".$input."'"
>blank column dropdown search to query all text fields
was kind of a throw-away. It may well be impractical. But I will likely do a similar 'task customization' and 'English field names' as above. I don't know Radicore well enough yet to comment on ease of these customizations. It's possible I'll write a different presentation layer, and possible my client may reject my suggestion of Radicore.

My intended point was about user expectations. I recently read a line about 'intuitive' software: "if your first guess works every time, it's a great piece of software." My first guess failed, so I noted it--if you want to remember it, write it down--so that I would change the software behavior before release to my own users. I remember liking parts of this short essay on web usability, Don't Make Me Think.

Subject: Re: Development Diary (markcarranza)
Posted by markcarranza on Thu, 20 Dec 2012 03:53:01 GMT

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.

Subject: Re: Development Diary (markcarranza)
Posted by AJM on Thu, 20 Dec 2012 08:44:58 GMT
View Forum Message <> Reply to Message

a) The field names can be displayed with whatever translations you want in as many languages

as you care to support. This is all explained in the documentation.

b) Different people seem to expect different behaviour, and as it is impossible to satisfy everybody's expections I shall stick to the expectations which I have experienced in my own long career, and the expectations of those clients who I currently support. I do not see the point of making a change for a "new kid on the block" if it means changing the behaviour which is known by all my existing clients. I do what is best for the majority, not the minority.

Subject: Re: Development Diary (markcarranza)
Posted by AJM on Fri, 21 Dec 2012 08:16:32 GMT
View Forum Message <> Reply to Message

1) If you want your new application to have the same look and feel as your old application then I fear that Radicore will not be the right framework for you. Radicore is for building web-based applications and therefore uses HTML screens, so it will never have the same look and feel as a legacy desktop application. There may be certain things which you can change with javascript, but that depends on your skills with javascript. The purpose of Radicore is to provide functional screens as quickly as possible with as many facilities as possible. You can then customise the screens to your heart's content. But if you spend 5 minutes building a basic screen with Radicore followed by 5 weeks of customisations then I fear that the benefits of Radicore will be wasted.

2) Your comments on "List Task (Process)"

a. Too many buttons. Response: There are 14 navigation buttons in that particular screen for the simple reason that there are that many child screens to which it is possible to navigate. Different screens will have different numbers of navigation buttons.

b. Way too busy visually, hard to read, harder to find correct button. Response: It gets easier with time.

c. Buttons should be grouped visually by category of task. Response: You can easily change the order in which the buttons are displayed refer to the Sort Sequence field in Update Navigation Button

d. Button locations should not change when screen re-sizes. Response: I'm afraid you do not understand how HTML works. Each HTML document is automatically resized when the dimensions of the browser window change. This may mean that some controls are resized or even repositioned. This is a "feature" of HTML, and there is very little you can do to change it.

e. User-setting-optional tool tips should concisely describe function of each button. Response: It may be possible to provide tool tips by using the HTML "title" option on input controls, but whether they work or not will depend on the browser which you are using. I will add the task's description as a "title" attribute in the next release.

f. 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. Response: There are 3 buttons labelled "Navigation Button" in "List Task (Process)" because there are 3 ways in which

the MNU_NAV_BUTTON table can be viewed and maintained using different transaction patterns. Button labels should not be too long otherwise they will take up too much room on the screen. Besides, with the addition of tool-tips longer labels should not be necessary.

g. 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. Response: Different people seem to have different opinions as to which buttons should be where depending on how they are classified, and even then they cannot agree on what those classifications should be. In my opinion having different areas for different classes of buttons would take up too much space on the screen. The only compromise I have agreed to implement is to split navigation buttons into two types those on the top line do not require any pre-selection of rows in the data area while those in the bottom line do.

3) Likely need to edit core files, Radicore upgrades would need programmer interaction. Response: I repeat, Radicore is a tool for developers, not end users. If an upgrade to Radicore is released there is no need to upgrade an existing application unless it fixes a bug or it adds functionality that is required. In any case, it will be up to the developer to implement and test the upgrade on a development system before it is applied to the live system. It has been my experience that if an organisation without its own IT department hires a third party to develop an application for them then the wisest move is to take out a support contract with that third party for ongoing maintenance. Not having a support contract and then trying to hire other contractors on an ad-hoc basis to deal with urgent issues sounds like an option that is fraught with problems they would not know the application and would require a learning curve which would take time and would therefore be expensive.

4) With Radicore I might, for example, need to change user_IDs from VARCHAR to INT. Response: I can think of no sensible reason why you should ever want to do such a thing. In would be an enormous effort for no tangible benefit.

5) I do not know, but prudence tells me to anticipate need to significantly change many core Radicore files. Response: If you think that you may want to change any core files within Radicore then Radicore is probably not the right tool for you. Besides, if you change any core files I cannot be held responsible for the consequences and I will not be able to offer any support.

6) Upgrading to a new Radicore version, under this scenario, would require significant effort and an above-average PHP coder. Response: You have previously stated that you are in the process of writing a critical medical application that requires strict compliance with Federal regulations, so the idea that you would contemplate using anything other than above-average programmers I find rather surprising.

7) This is a medical-services client, the new software will contain highly confidential patient information. Response: The contents of any application database you create, and any code which you create to deal with that data will be your responsibility as your application indeed *ANY* application which has been built using Radicore is completely unknown to me and outside of my control. If you or your client are worried about the security capabilities of Radicore all I can do is point you to The Radicore Security Model

8) Imagine changing all existing Radicore PDFs to web pages, with (optional) print button and

print css. Response: You don't change PDF documents to web pages. A web application contains HTML documents, and HTML documents cannot be used to create large reports which can be sent to a printer. The client can use the browser's "print" button to print what is currently being displayed in the browser window, but that's not the same thing. With Radicore you can produce web pages containing whatever information you like. You can extract that data to a CSV file, but you can't send it to a printer which is attached to the client device.

9) I'd love to see a system allowing user-created queries and reports. Response: Then what you want is a customisable report generator which you can obtain from a third party and which connects directly to the database, thus bypassing any "limitations" within Radicore.

10) Haven't installed example systems yet, it would be great if one showed significantly different look and feel. Response: You'll be disappointed then, as all the example systems share the same look and feel. This is because they all share the same transaction patterns, the same XSL stylesheets, and the same CSS files. It is this sharing of standard and reusable components which allows Radicore to be a Rapid Application Development toolkit. If you want everything to be different and unique you can kiss goodbye to sharing common components and kiss goodbye to rapid application development.

11) User Interface. Response: If your client is used to a Windows GUI in a legacy application then they will disappointed that a modern web application is not exactly the same. A web application uses HTML documents for its screens, and HTML (with CSS) can look different in different web browsers. If your clients are unwilling or unable to accept the differences in a web application I can only suggest that you stick with a desktop application.

12) Quick Search bar. Response: I have changed the button text from plain "Search" to "Quick Search" to differentiate it from the other "Search" button.

13) "Locked" checkbox. Response: Like yourself I thought that the default for this control should be ON, but my major client disagreed and insisted that it be turned OFF. It is impossible to please everybody all of the time, so I am afraid that somebody will always end up being disappointed that they cannot have their own way with everything.

Subject: Re: Development Diary (markcarranza)
Posted by markcarranza on Sun, 30 Dec 2012 01:28:36 GMT
View Forum Message <> Reply to Message

Again, thank you for replies.

1) If you want your new application to have the same look and feel...
Not the same, but ... a rich web UI. I'm used to (framework guy) spending 80% of development time on user interface.

There are many things I like about Radicore architecture, especially business layer + workflow, and data access layer + audit. I build highly-customized Rich Internet Applications (RIAs) with JavaScript.

I think I can replace the presentation layer, using a similar small-component philosophy, and still reap some benefits of the Radicore back-end, but it will be a risk. If you think this is an unwise risk, please let me know.

2)d. >resize, reposition
easy to fix location with css. It's my experience that webapp users prefer 'fixed locations and scrolling' rather than 'scrunching.' The interface of this forum's post form is a good example: some things reposition intelligently with resize, but most fixed.

4) With Radicore I might, for example, need to change user_IDs from VARCHAR to INT.
>no sensible reason
For the nonsensible reason that (major client) employees want to change their login name to reflect name changes. So I separate the string used for logins (login_name) from the primary key (user_id, employee_id).

For the sensible reason that users are (typically) employees, and while not all employees are users, using employee_id in 'created_user' and 'revised_user', as well as other application 'user' fields, keeps joins sane in applications with many employee/application integrations. For enterprise clients I use employee_id (technical key) as the primary key of an application user table, the login name is a separate, unique VARCHAR.

7)>The Radicore Security Model
Thank you for the link

9) I'd love to see a system allowing user-created queries and reports.
>customisable report generator which you can obtain from a third party
I haven't found a good one for HTML reporting. They typically have data dictionaries...
Suggestions welcome.

In the past I've made a number of custom print-related plug-ins, but all browser specific, IE or Mozilla. Print requirements are still TBD.

Subject: Re: Development Diary (markcarranza)
Posted by markcarranza on Tue, 01 Jan 2013 01:54:08 GMT
View Forum Message <> Reply to Message

A key realization for coding with Radicore: a task file has code like
$table_id = 'dict_column'; // table name
or
$outer_table = 'dict_table' // name of outer table
$inner_table = 'dict_column'; // name of inner table
but these variables aren't exactly names of tables.
They're class names of subclasses of the business object, Default_Table, or further subclasses, found in /[subsystem]/classes/[name of table class].class.inc
class dict_column extends Default_Table
// or

class dict_column_s01 extends dict_column
I could change the default transaction pattern code to read like:
$table_id = '#tablename#'; // name of table subclass
or
$outer_table = '#outer_table#' // name of outer table task subclass
$inner_table = '#inner_table#'; // name of inner table task subclass

I've started my own naming convention using a 'task abbreviation' like so,

task file = [table]([transaction_pattern])[task abbreviation].php
screen file = [table].[transaction_pattern].[task abbreviation].screen.inc
report file = [table].[transaction_pattern].[task abbreviation].report.inc

So I'm contemplating simply generating with new tasks, task-specific business classes,

class file = [table]_[task abbreviation].class.inc
require_once '#tablename#.class.inc';
class #tablename#_#task_abbrev# extends #tablename#and including all(?) foundation '_cm_' method stubs with terse comments and/or a comment link to documentation.

Generating a task file as,

$outer_table = '#outer_table#_#task_abbrev#' // name of outer table-task subclass
$inner_table = '#inner_table#_#task_abbrev#'; // name of inner table-task subclass

For basics, no changes to the subclass, but having an empty dbObject pattern tied to the task would, I think, more than balance the bother of extra files that basically do nothing. Especially for newbies to Radicore. Maybe it could be turned off for experts...

Subject: Re: Development Diary (markcarranza)
Posted by AJM on Tue, 01 Jan 2013 12:28:52 GMT
View Forum Message <> Reply to Message

1) I'm used to spending 80% of development time on user interface.

That's where we differ. I spend 80% of my time on functionality. I've given up trying to make things "look pretty" as everybody has a different opinion as to what "pretty" means. If you want to change how the RADICORE screens look then you don't really have to replace the whole of the presentation layer - you can drop in your own CSS files, and you can add JavaScript if you want to. Those facilities have been built into the framework, so it is up to you if you make use of them.

2) It's my experience that webapp users prefer 'fixed locations and scrolling' rather than 'scrunching.'

I am only interested in the default behaviour of browsers, which is that the document will expand and contract to fit the current window. If certain users expect something different then they will be disappointed as I am not going to do anything to override the browser's default behaviour, which I

probably couldn't do anyway.

3) With Radicore I might, for example, need to change user_IDs from VARCHAR to INT.

You don't need to. If you wish to link a VARCHAR user_id to an INT on another system they why don't you make use of the PARTY_ID column on the MNU_USER table? This is what I use to link users to parties on my PARTY database.

4) HTML reporting

There is no such concept as an HTML report. An HTML document is designed to be rendered in a browser and cannot be sent directly to a printer. There is an option in the browser to print the current document, but that is controlled from the client, not the server.

If you want to give users the ability to generate their own custom reports from your own application database then you will have to use a 3rd party report generator. I cannot recommend any particular one, but just do a google search on "report generator" and you will see a huge number of results. Just take your pick.

Subject: Re: Development Diary (markcarranza)
Posted by AJM on Tue, 01 Jan 2013 13:18:55 GMT
View Forum Message <> Reply to Message

1) these variables aren't exactly names of tables.

They are the names of class files, which usually are related to database tables. Names such as "dict_column_s01" identify subclasses which contain task-specific behaviour.

2) I could change the default transaction pattern code to read like ...

You could, but you would be wasting your time.

3) I've started my own naming convention

Another waste of time.

If you don't like Radicore's naming conventions then don't use Radicore. If you try to change the core code to deal with your naming conventions then you will end up expending a great deal of effort for abolutely no benefit, and you will get no support from me.

Subject: Re: Development Diary (markcarranza)
Posted by markcarranza on Tue, 22 Jan 2013 03:15:44 GMT

View Forum Message <> Reply to Message

Fatal Error: Some data has already been output, can't send PDF file (# 256).

the problem was: extra "\r\n" in .class.inc file AFTER closing PHP tag '?>' (two extra lines - http://php.net/manual/en/language.basic-syntax.instruction-s eparation.php)

I typically leave off the closing tag in files that are loaded with 'require_once', 'require'

Subject: Re: Development Diary (markcarranza)
Posted by AJM on Tue, 22 Jan 2013 06:59:09 GMT
View Forum Message <> Reply to Message

Which *.class.inc file?

Subject: Re: Development Diary (markcarranza)
Posted by markcarranza on Tue, 22 Jan 2013 16:15:11 GMT
View Forum Message <> Reply to Message

>Which *.class.inc file?
One of my own new .class files. My mistake.