

The Evolution of the Role of the ProAdvisor: From Bookkeeping and QuickBooks Training to Working With, and Becoming, a Developer – Part II

Last month I talked about the initial stages of the evolution of the role of the ProAdvisor – discovering the “need” for automation, then the “search” for that automation, and the time-consuming “work-a-rounnds” that we develop in order to obtain “some” automation. If you missed that article, you can find it in the Newsletter Archive section at <http://www.idnac.org/news-archive.html> in the May Newsletter.

This month I’ll talk about what it is like to work with a programmer (aka Developer) and my own evolution from Bookkeeper/ProAdvisor to Developer. While this “evolution” may not be for you, hopefully what you will take away is a better understanding of “**how**” a developer thinks, which can only benefit your own interaction with Developers of QuickBooks compatible software.

How a Developer thinks....

I think the first thing to discuss is “**how**” a developer thinks – because they do think quite differently than a Bookkeeper/ProAdvisor; and this was perhaps the hardest lesson that I had to learn! Let me explain....

As Bookkeepers/Advisors we look at a report and everything is arranged in an orderly fashion; with columns of numbers that “**should**” all add up; therefore, we tend to think in an orderly “black & white” fashion.....“either it adds up or it doesn’t” and if it doesn’t we **then** need to figure out why.

A Developer, on the other hand, thinks first about “**how to make it all add up**” what mathematical formulas and lines of code that they will need to write in order to gather all the data from various locations and add it up in order to output it in it’s final format...the one that we see and understand; in other words they think in “fuzzy logic” and definitely “outside the box”.

These differences in “thinking” patterns can cause all sorts of issues....frustration, miscommunication, and simple overall dissatisfaction with each other.

Let me give you an example of a conversation that Ben & I had over the “way I thought” that our certified payroll program should work, it went something like this....

“Really it should be pretty easy. All you need to do is reach into QuickBooks and read information from timesheets, employee records, job records, paycheck transactions and you can get this, this, and this.....then build “**something**” like a database that will contain linked records back to these items which will hold these pieces that QuickBooks has no way of tracking but that are required on the final reports. Oh yeah, and you need to figure out a way to merge all this information and put it in “**this**” format”, one for each job...at which point I handed him an Excel template of one of about 50 different certified payroll report formats.....

Sounds fairly straightforward.....doesn’t it? ☺

The resulting conversation was not pretty (even the dog covered his ears)! There was a lot of "it's just NOT that simple" (Ben) with "YES, it is" (me). This went on for nearly a week as I heard him typing feverishly.

Finally in desperation to make me "understand" he handed me a 6 page typed document and said "here, this is JUST the code that's required to access QuickBooks, get the timesheet information, and begin to parse it out".

My first reaction – "you've got to be kidding" and THEN I started to "attempt" to read what he'd handed me! It looked like "gibberish" there was few if any "real" words. I should also mention; I had to ask him what 'parse it out' even meant! (For those of you that are wondering about what "parse it out means" - my best "non-technical" explanation is: dividing or separating large amounts of information into smaller common pieces). That was my first encounter with "code".....

Today the "code" for our software is easily more than 1,000 printed typed pages (that's both sides, mind you) and every once in a while I'll sit at his computer and "look" at it in my attempt to "understand" what all goes on behind the scenes to make it do what I thought was **so simple**; and my reaction is still one of "this is worse than Greek hieroglyphics".

The important things that I want to share with you are:

- While something **may sound** simple; from a coding or programmatic standpoint it's not
- While it can take 1,000's of pages of typed code to make something work – all it takes is just **ONE** typo to break it

This leads to our next topic...

Working with a Developer to create QuickBooks Compatible Software....

Over the years I've heard a lot of Bookkeepers/ProAdvisors talk about how important "billable time" is; me included...after all, as Advisors, our income depends upon the ability to "bill the time we spend on performing various activities for our clients". Our income is solely derived from "selling" our knowledge about a task and is based in increments of "how long" it takes to complete that task at an hourly or per service rate. As a Bookkeeper/Advisor if we spend 10 hours performing a specific task for a client we can bill that client for 10 hours at a predetermined rate.

"It's just NOT that easy" for a developer....

Developing and creating software is a highly underpaid, and often risky, venture for Developers. Many software programmers have been down the "development road" before; many times with tragic results.

- They have had code stolen
- Never been paid for their coding time
- Been offered the opportunity that they just couldn't refuse because there was a "huge market" for this type of product to only sell 1 installation of the product
- Have to hire and pay another programmer in order to meet production deadlines while not receiving any money themselves

Being a developer is far different than being a Bookkeeper/Advisor because if you perform services for a client and that client doesn't pay you, you have the luxury of refusing to do any additional work for that client until the pay you. And relatively speaking your risk is far less.

Developers do not see any income until **AFTER** the product is completed and they are able to actually begin selling the product. But it really involves so much more than that simple statement, and to give you an idea of just how much more there is, I'll share my own experiences with you.

It took Ben and I each – 8 months of working 30-40 hours a week (we each had jobs in addition to this) to develop our certified payroll program. We started the development process in August of 2000 and made our first sale in March 2001. That was approximately 36 weeks with 1080-1440 hours each or a total of 2160 – 2880 hours without seeing even one penny in revenue. Even at a modest rate of \$35.00 per hour that was a total of \$75,600.00 to \$100,800.00 in “unbillable time”.

Could we sell our software for that price? Absolutely not!

With its initial release our software would produce a certified payroll report, statement of compliance, “no work” performed payroll and 2 EEOC Reports. We initially released it by “state” reporting requirements – so that meant that there were 22 “different versions” and decided on a selling price of \$349.00 per company license.

This meant that we needed to sell 216 – 288 packages just to cover our initial time.....and that’s just the very basics of things and doesn’t include the overhead or additional “unbillable hours” that it will take to make those initial sales (marketing, building a website, documentation, etc.). And our situation was probably very unique, because we didn’t have employees that we had to pay during the development stage, because it was just us and our time.

The important things to take into consideration when asking a developer to create a product:

- Realize that they aren’t going to “jump” at the opportunity to create a QuickBooks compatible application.
- Realize that there will be a huge difference between your amount of unbillable time and their amount of unbillable time.
- Realize that developers may ask questions like:
 - What do you think I can sell this product for?
 - How many products do you think I can sell?
 - How many of these products can **you** sell?
 - How can I or you market this product?

These are just some of the questions that may be asked by a developer when they are approached to create a QuickBooks compatible application.

As Bookkeepers/ProAdvisors we may not appreciate being asked these questions, but they are legitimate questions and frankly you should expect them **and you should answer them as honestly as possible** when they ask you what you think the product could sell for, how many you think could be sold, how many products you could sell, and how the product could be marketed. Realize that the Developer is not looking for a “personal guarantee” from you; they are just seeking information before making any sort of commitment to a project.

That’s it for this month; hopefully I’ve given you some food for thought – so to speak. Next month we’ll talk about what is involved in the creation of a QuickBooks compatible add-on; which will help you to understand “**why**” developers will ask these types of questions and “why” you should answer their questions as honestly as possible.

Until then

Nancy