

This is an excerpt from the
2011 QuickBooks Consultant's Reference Guide.

[Click here for more information.](#)

Copyright © 2010 The Sleeter Group – www.sleeter.com

create a table of contents for a specific document - and you don't otherwise use Word much - this might have merit for your pocketbook.

Cloud Computing in Context

Do we throw away the desktop model because cloud computing is the IN thing? There are many challenges yet to cloud computing. Most of which revolve around security and backup procedures, but also revolve around more technical challenges like data rollback, or data auditing; the latter two are VERY difficult, and keep bright minds up late at night.

Let's look at some other challenges. First, there are millions of users still comfortable keeping their data and applications right where they are now - on the DESKTOP.

Second, reproducing the desktop features of an application is not always possible - or easy. Think of the years it took to improve products like Quicken and QuickBooks, with year after year of added features. You can't reproduce that functionality in an internet application at the drop of a hat; some features might be impossible to reproduce.

Third, what about the processing power? If you have an application that is built in the cloud, it is sharing processing power with hundreds, or even thousands of other users. To type words on a piece of "cloud paper", won't be a big deal; using Photoshop to render thousands of pixels would be.

Finally, we need to remember the psychology of the generations. The younger generation is much more comfortable putting their whole life in pictures up on the web. Many members of the other generations wouldn't consider putting their financial data in the cloud - never mind pictures of their last night on the town.

Understanding IPP

We better get used to it. The IPP is here, and it is a great improvement over the SDK.

Intuit has invested in cloud computing in a STRONG way, by creating the IPP Platform. Loosely modeled after the Salesforce.com developer cloud implementation, the IPP platform provides a simple and easy mechanism for developers to create- and sell - internet applications that interface directly with QuickBooks.

The IPP platform contains tools for database creation, programming, billing, usage metering, licensing procurement, security mechanisms, and even creates a listing for the product that can be accessed directly from within QuickBooks. In the desktop world, this would be paramount to nirvana.

I have sat in several roadmap meetings, and my first notion is that they have learned from their experiences in the SDK implementation, and have provided a top notch environment for developers to create a sellable application in a short timeline. In fact, using the IPP platform and tools, a developer can bring an application to market in weeks, instead of months.

As good as the SDK was for creating rules for importing data into QuickBooks, it was still left to the developer as to how error checking was assessed BEFORE attempting to import data. A

good example of this was duplicate invoice numbers, duplicate customers, or duplicate sales transactions. Smart developers would query the QuickBooks data to ensure that duplicates would not exist; but not all developers followed this method - or even knew how to.

My friend and Intuit colleague Peter Vogel created a great tool called the *web connector* which is the current standard for importing data into QuickBooks from shopping carts or online applications, but again, unless the developer brings the QuickBooks data from the desktop into the shopping cart database to check for duplicates, you would have to periodically check your QB data.

The IPP overcomes this limitation, because all of your QuickBooks data is brought up into the cloud by default - and all error checking, and data synchronization, is done IN THE CLOUD. I personally welcome this improvement, and I strongly applaud the Intuit implementation of this process.

Does the IPP replace the SDK?

Intuit is adamant that the SDK will continue to be updated for those developers who maintain and create desktop applications - although having said that, there is no new SDK for 2010 or 2011. But for those who want to create internet applications that extend the capabilities of QuickBooks, the IPP is the way to go.

Get this: Intuit even handles the billing of your application to your customers? (Of course Intuit gets a piece of the action, but we will describe that more in detail below).

Cloud computing - easier to implement, but with some caveats

Cloud Computing is here to stay and the SDK offering is not the appropriate mechanism to integrate QuickBooks data with other Cloud - or SaaS - Applications. So Intuit devised a more fluid mechanism that would allow developers to inculcate QuickBooks data into their cloud applications.

At first glance we know that SaaS applications are much easier to write than desktop applications. Developing a desktop application requires a developer to accommodate different operating systems, multiple patches, and interoperability with other applications on a user's PC. Desktop application patches and version updates require lengthy cycles that can lead to massive rollouts, help desk costs, and end user headaches

However, there are a few new wrinkles. Instead of multiple desktops and installations, the developer will use a hosting platform - including powerful hardware, server operating systems, database applications, programming tools, network equipment and metering tools, and a place where the equipment will be housed.

Whereas the desktop/server application leaves the basic security in the hands of the end user, the security for the cloud application is now the responsibility of the developer hosting the data.

How developers host their Cloud Applications

Developers can use one of the following options to make their application available to you.

The first option is to host their own server farms, while handling all of the network routing and security issues. This is rare, and can be very costly for a small developer; think of the investment needed for all of the hardware, internet pipelines, network gear, etc. Let's face it developers are in the business of developing - not becoming system administrators.

The second option is to rent the server and network equipment, data center space. This eliminates the need for investing in costly hardware and appropriate warehousing, but the

developer still needs to setup and administer the server software, database structure, user metering mechanisms, security measures, and programming tools. Again the developer wants to develop - not administer.

In addition the developer would have to build a self-contained billing component - which are costly, and difficult to implement because of security regulations and credit card privacy concerns.

Renting the hardware, space, etc. is commonly known as IaaS or Infrastructure as a Service. The third option is to find a company that provides ALL of the components e.g. hardware, software, data space, database administration, billing infrastructure, a variety of programming tools, etc. For the developer this is really the optimal method - because the developer gets to DEVELOP, and can focus on shorter cycles of application development.

Whereas desktop application cycles can run for months or even years, a Cloud Application can be six to eight weeks. This method is called PaaS.

So what does the IPP do for the developer?

The IPP offering is a PaaS offering, and designed with basically one thing in mind: Let the developer DEVELOP.

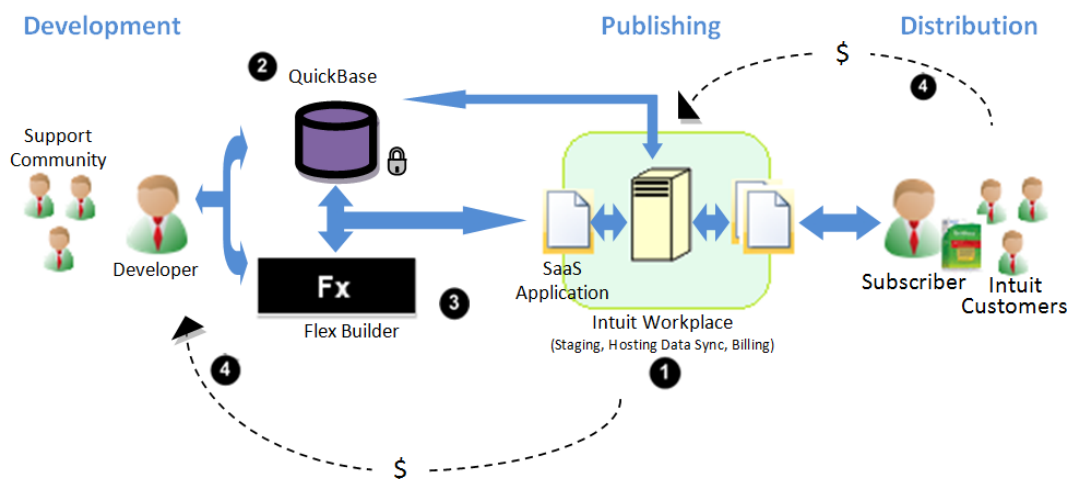


Figure 14-23 IPP Flowchart (based on image by Intuit)

Figure 14-23 is a re-creation of an Intuit illustration, but with our own numbering system; it delineates the pieces of their offering. Let's describe some of these components in detail.

1. **The Intuit WorkPlace**

Intuit hosts the application, provides access to the QuickBooks data for integration and synchronization purposes, and provides a billing mechanism. This is the core of the offering.

2. **QuickBase**

For those of you who know what QuickBase is, you are probably wondering "what is QuickBase doing here". For those of you who don't know what QuickBase is, it is an online database offering from Intuit, and is used by Fortune 100 business and small business alike. The IPP offering makes available the database functionality of QuickBase - in raw form - to those developers who need a database mechanism within their application; and charges separately for its use.

3. Flex Builder

This is a cross-platform (can be used on windows or linux servers) framework made available by Adobe, that allows developers to rapidly build web applications that - according to Adobe - can be deployed “consistently on all major browsers, desktops, and operating systems.” Think of it as a programming tool with a toolbox of little building blocks which already contain pre-programmed instructions. Companies like The American Cancer Society, and the Discovery Channel are just a couple of well-known entities who use Flex for their web development, because of its power, simplicity, and ease of use.

4. Billing System

All applications that avail themselves of the Intuit WorkPlace are billed directly from Intuit to the customer. In turn, Intuit uses a formula that includes things like revenue share, database usage fees, taxes, refunds, etc. and deposits the developer share into their account at the end of each month. Revenue share can be as high as 20%, and database usage fees rely upon the amount of data stored in QuickBase, and the amount of time that someone spends accessing that data. If you really want to try and understand ALL of the exact details, you are on your own - I never got the rocket science degree that someone needed to come up with the appropriate formulas. If your curiosity is that overwhelming, visit https://ipp.developer.intuit.com/Get_Paid/Revenue; you will see what I mean.

What Intuit DOESN'T do, is determine what the developer charges to the end user. The developer has to assess what the potential usage fees might be (using a slide rule and calculator); accurately forecast their potential subscriber base, and then come up with a fee that will cover all of those charges - including Intuit's cut - and find a profit margin suitable to the business. (For all of you number crunchers out there, there is potential business in developer land).

Existing SaaS applications

So what happens if a developer/company already has a SaaS application hosted somewhere else?

Intuit recognized this need early on, and provided a means for existing SaaS applications to connect the IPP Intuit WorkPlace and avail themselves of two fundamental elements: QuickBooks data for integration, and access to the millions of QuickBooks users who may potentially use their product. They are called *Federated Applications*.

Currently there are over several dozen federated apps availing themselves of the IPP offering. MyFax, ExpenseWare, and Pixily are several of the more recognized names in this group.

The IPP offering provides capabilities for these federated applications to “publish” their application, thereby making it available to the Intuit end-user community - just like any other IPP based application.

Benefits

What are the real benefits to a developer wanting to use the IPP platform?

- **The Intuit Workplace**
It provides the hardware, server operating system, billing mechanism, AND security, without the expensive up-front investment.
- **QuickBooks data**
The method for synchronizing the QuickBooks data in the cloud is a far superior mechanism to the SDK. As of this writing, not all of the QB data is available in the