Does anyone have database, programming language/framework suggestions for a GUI point of sale system?

Asked 16 years, 2 months ago Modified 10 years, 8 months ago

Viewed 3k times

7

Our company has a point of sale system with many extras, such as ordering and receiving functionality, sales and order history etc. Our main issue is that the system was not designed properly from the ground up, so it takes too long to make fixes and handle requests from our customers. Also, the current technology we are using (Progress database, Progress 4GL for the language) incurs quite a bit of licensing expenses on our customers due to mutli-user license fees for database connections etc.

After a lot of discussion it is looking like we will probably start over from scratch (while maintaining the current product at least for the time being). We are looking for a couple of things:

1. Create the system with a nice GUI front end (it is currently CHUI and the application was not built in a way that allows us to redesign the front end... no layering or separation of business logic and gui...shudder).
2. Create the system with the ability to modularize different functionality so the product doesn't have to include all features. This would keep the cost down for our current customers that want basic functionality and a lower price tag. The bells and whistles would be available for those that would want them.
3. Use proper design patterns to make the product easy to add or change any part at any time (i.e. change the database or change the front end without needing to rewrite the application or most of it). This is a problem today because the Progress 4GL code is directly compiled against the database. Small changes in the database requires lots of code recompiling.

Our new system will be Linux based, with a possibility of a client application providing functionality from one or more windows boxes.

So what I'm looking for is any suggestions on which database and/or framework or programming language(s) someone might recommend for this sort of product. Anyone that has experience in this field might be able to point us in the right direction or even have some ideas of what to avoid. We have considered .NET and SQL Express (we don't need an enterprise level DB), but that would limit us to windows (as far as I know anyway). I have heard of Mono for writing .NET code in a Linux environment, but I don't know much about it yet. We've also considered a Java and MySql based implementation.

To summarize we are looking to do the following:

1. Keep licensing costs down on the technology we will use to develop the product (Oracle, yikes! MySQL, nice.)
2. Deliver a solution that is easily maintainable and supportable.
3. A solution that has a component capable of running on "old" hardware through a CHUI front end. (some of our customers have 40+ terminals which would be a ton of cash in order to convert over to a PC).

Suggestions would be appreciated.

Thanks

[UPDATE] I should note that we are currently performing a total cost analysis. This question is intended to give us a couple of "educated" options to look into to include in or analysis. Anyone who could share experiences/suggestions about client/server setups would be appreciated (not just those who have experience with point of sale systems... that would just be a bonus).

[UPDATE]

For anyone who is interested, we ended up going with Microsoft Dynamics NAV, LS Retail (a plugin for the point of sale and various other things) and then did some (and are currently working on) customization work on top of that. This setup gave us the added benefit of having a fully integrated g/l system, which our current system lacked.

database frameworks client-server point-of-sale

Share

Improve this question

Follow

edited Apr 15, 2014 at 14:43

asked Oct 10, 2008 at 15:40

Jason Down

22.1k ● 13 ● 87 ● 118

5 Answers

Sorted by: Highest score (default)

2

Java for language (or Scala if you want to be "bleeding edge", depending on how you plan to support it and what your developers are like it might be better, but also worse)

H2 for database

Swing for GUI

Reason: Free, portable and pretty standard.

Update: Missed the part where the system should be a client-server setup. My assumption was that the database and client should run on the same machine.

Share Improve this answer Follow

edited Oct 10, 2008 at 16:52

answered Oct 10, 2008 at 15:51

John Nilsson
17.3k ● 8 ● 35 ● 42

Any chance you can give some insight into why you are suggesting these? Is there a specific advantages to using these over something else? Thanks – Jason Down Oct 10, 2008 at 15:54

Java is a standard as it comes. And it's portable, which you kind suggested it should be. H2 is an embedded database, which should be helpful in a specialized application such as yours. Also H2 (by the creators claim) is performant and light. Swing just to avoid dependencies. It's good enough. – John Nilsson Oct 10, 2008 at 16:50

1

I suggest you first research your constraints a bit more - you made a passing reference to a client using a particular type of terminal - this may limit your options, unless the client agrees to upgrade.

You need to do a lot more legwork on this. It's great to get opinions from web forums, but we can't possibly know your environment as well as you do.

My broad strokes advice would be to aim for technology that is widely used. This way, expertise on the platform is cheaper than "niche" technologies, and it will be easier to get help if you hit a brick wall. Of course, following this advice may not be possible if you have non-negotiable technology already in place at customers.

My second suggestion would be to complete a full project plan, with detailed specs and proper cost estimates, before going with the "rewrite from scratch" option. Right now, you're saying that it would be cheaper to rewrite the system than maintain it, and you don't really know how much it would cost to re-write.

Share Improve this answer Follow

answered Oct 10, 2008 at 16:04

Bork Blatt
3,368 ● 2 ● 21 ● 17

The rewrite is not a for-sure thing yet (I meant there is a current total cost analysis being performed right now, with the looks that we want to do a rewrite). We're investigating options for a rewrite which will inform more of the cost. That is what this question is for, to give educated ideas.
– Jason Down Oct 10, 2008 at 16:12

BB hit the nail on the head about picking a prolific technology. It really sucks when your one expert in SupaDupaDB quits and you can't find another for all the money in China.
– Mark Brady Oct 10, 2008 at 17:46

1

I suggest you use browser for the UI.

Organize your application as a web application.

There are tons of options for the back-end. You can use Java + MySQL. Java backend will save you from windows/linux debate as it will run on both platforms. You won't have any licensing cost for both Java and MySQL. (Edit: Definitely there are a lot of others languages that have run-times for both linux & windows including PHP, Ruby, Python etc)

If you go this route, you may also want to consider Google Web Toolkit (GWT) for creating the browser based front-end in a modular fashion.

One word of caution though. Browsers can be pesky when it comes to memory management. In our experience, this was the most significant challenge in doing browser based POS You may want to checkout Adobe Flex that runs in browser but might be more civil in its memory management.

Share Improve this answer Follow

answered Oct 10, 2008 at 15:52

Tahir Akhtar
11.6k ● 7 ● 45 ● 69

I can't really see how the added complexity of using HTTP and friends between the GUI-layer and the system will benefit anything. – John Nilsson Oct 10, 2008 at 16:04

Be careful with browser based UI when developing Point of Sales system. It usually integrate with specialized hardware (e.g. Thermal Printer, Barcode Scanner, etc.) – Snackmoore Mar 19, 2010 at 4:13

1

What is CHUI? Character-UI, as in VT terminals? Or even 3270 style?

It sounds like you need a 3-tier system - the database backend, a middle-layer that runs the bulk of the back-end business processes, and a front-end layer for the CHUI / GUI / data-gateway.

All three layers can reside on one machine; or you can distribute the tiers out to various servers. The front-end layer would control the actual terminals, whether they are VT-terminals, or a web-browser, or a custom-written 'client' application.

Make sure you have considered the hardware needs here -- are you going to have barcode scanners, cash drawers, POS debit/credit terminals, et cetra? If you are using a standard browser, it might be hard to reliably integrate those items. (At the very least, you're likely going to have to write special applets to handle them.)

Finally, consider the possibility of a thin-client technology on Windows. It greatly simplifies system management, since you only have to upgrade the software centrally. Thin-client PC's are cheap -- sub $200.

Share Improve this answer Follow

answered Oct 10, 2008 at 17:14

Toybuilder
11.5k ● 5 ● 30 ● 34

1

Golden Code Development (see www.goldencode.com) has a technology that does automated conversion of Progress 4GL (the schema and code... the entire application) to a Java application with a relational database backend (e.g. PostgreSQL). They currently support a very complete CHUI environment and they do refactor the code. For example, the conversion separates the UI, the data model and the business logic into separate Java classes. The entire result is a drop-in replacement that is compatible with the original (users don't need retraining, processes don't need to be modified, the data is migrated too). This is possible because they provide an application server and a set of runtime classes that provide that compatibility. The result of the automated conversion is not something that needs further editing before you can compile and run it. True terminal support is included so hardware terminals still work (it requires a small JNI library to access NCURSES from Java). All the rest of the code in the runtime is pure Java. No Progress Software Corp technology is used in the resulting system and it runs on Linux.

At least one converted system is already in production, running a 24 by 7 mission critical environment. It is a converted ERP system that their mid-sized pilot customer uses to run their entire business.

Share Improve this answer

Follow

answered Aug 26, 2010 at 18:04

Greg Shah

11 ● 1