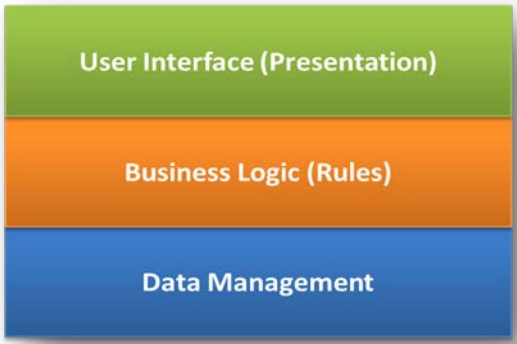
Process Automation from Scratch

Over the course of 2013 I have written about a number of canned off the shelf (COTS) products that can be used to automate processes with some degree of "configuration." Sometimes, configuring a COTS solution is just not going to get you there. In these cases, you're going to need to start with some sort of design and build what you need.

Software Basics

BPTrends has dedicated significant resources to reporting on business process management system (BPMS) tools. A closer look at products like Appian, IBM Websphere, or Oracle's BPM Suite reveals that these are essentially development frameworks. By this, I mean that you will need to employ a software development professional to build anything of significance.



When we talk about software development, it's easy to start getting complicated right out of the gate, but the reality is that all software has three basic layers of functionality in common. Let's break it down briefly.

Data Management

If you're going to build a business application (or almost any application), you need to manipulate some data. In order to do that, it must be stored someplace. We find ourselves dealing with ever increasing amounts of data. That may mean handling large numbers of transactions simultaneously. It could also mean performing business intelligence (BI) on massive amounts of data in order to distill it into a form that is useful for decision making.

The nature of data is also changing. Once upon a time, businesses dealt only with data entered into a structured form. Now, we must be prepared to deal with images, videos, web pages, freeform documents and audio. Traditional data has been best handled by a class of data handling tools known as Structured Query Language (SQL) or relational databases. Using these data storage mechanisms typically requires some rigor in predefining how data will be stored.

Many new large scale applications are looking at a different class of storage mechanism known as noSQL or document store. These tools allow for a more on-the-fly approach to structuring data for retrieval that makes them more flexible and easier to manipulate when the structure of data is dynamic or difficult to predefine.

Business Logic

This is the layer that BPMS tools attempt to simplify. Traditionally, this is where the programming code gets generated. Any way you slice it, someone needs to understand how to transform abstract ideas about how to handle various scenarios into concise instructions on which a computer can act autonomously. Whatever tool you use to make this happen, you will need to learn the “language” of setting up these rules.

The good news is that many of the “rules” we use to control how software behaves are predictable. Take, for example, required data. This is a rule that says, “You cannot leave this information blank.” This is such a ubiquitous type of requirement that many tools can anticipate this need and not require actual coding since it is a “yes or no” parameter (i.e. yes, this field is required or, no, it is not required). The people that design software development tools try to incorporate as many of these common business rules into the design interface so no coding is necessary.

User Interface

The User Interface (UI) the layer we the end-users are most familiar with. Some programs may just talk directly with other programs and pass data. These programs don’t have a UI, but most of the process automation tools you build will have some sort of human interaction and to do that, they will need a UI.

The good news is that there is an emerging standard here on which new development has focused. Since devices increasingly share web browsing capabilities, program interfaces should be designed for that environment. The standard for this is hyper-text markup language (HTML). The latest version of HTML is 5 (HTML5). This is an exciting advancement because HTML5 has introduced the ability to handle a wide array of data types and do so in a much more efficient manner. Before HTML5, developers needed to supplement their interface designs with proprietary components that required web browsers to be enhanced with add-ins, which were not available for all devices, to function with their application.

For modern devices HTML5 is all that is necessary. Some developers still use JavaScript, which is almost as ubiquitous as HTML5. Between these two technologies, there is little an interface designer can’t do inside of a web browser like Internet Explorer, FireFox, Safari, or Chrome.

Rapid Application Development (RAD)

“Rapid” is in the eye of the beholder. The typical horse drawn carriage traveled between 5 and 12 mph. To them, the Model T’s top speeds of 40 – 45 mph must have seemed blinding fast and today’s 70 mph freeway speed limits hard to imagine.

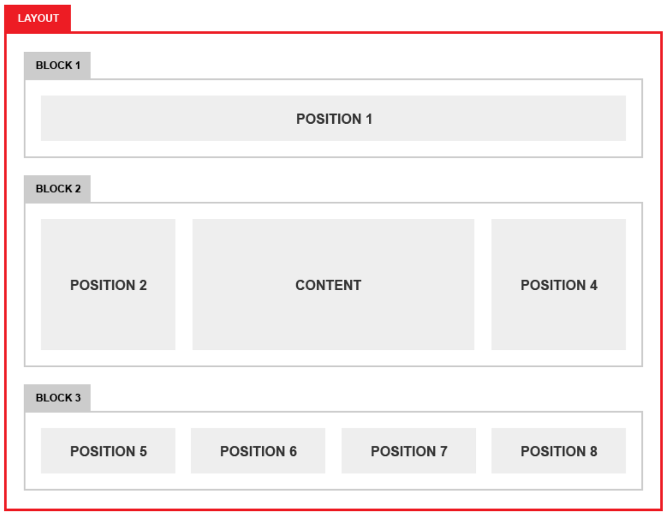
Application development has enjoyed a similarly continually adjusting definition of “rapid.” Whereas “miles per hour” provides a standard measurement of transportation speed, there is no such standard for measuring development speed. Alas, the complexity of the application, the tools being used, and the personnel doing the work all converge to determine the relative speed with which an application can be created.

At each layer of our model, software development tool makers have attempted to find ways to automate the process. Let's take a look at each layer to see what they've done.

Drag and Drop Screen Builders

Once upon a time, screens were built using code by placing text or input commands (e.g. text boxes, dropdown lists, and checkboxes) at specific screen coordinates. This was great when we could count on all screens being the same size. Obviously, this has become a bad assumption to make. Eventually, this evolved to a "drag and drop" interface that allowed users to place the screen objects selected from a visual pallet onto the screen and link them to the data they represent in the data layer.

These UI builders come with basic Create, Read, Update, and Delete (CRUD) functionality. They also can have Sort, Search and Filter capabilities for data in a table or grid format in which multiple records are displayed in rows on the same screen (i.e. a list).

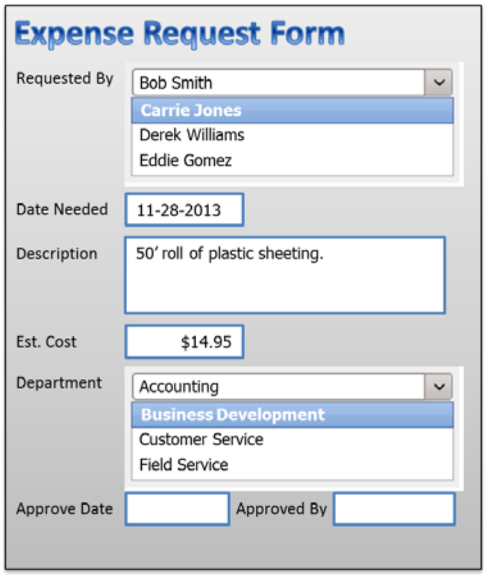


One important feature of a modern screen builder is that the screens it builds must be "responsive" (or what some call adaptive). This means that the layout will change dynamically based on the size of the screen. For example, tabs that appear across the screen of a tablet or laptop automatically turn into a dropdown list on a smartphone. Input fields will automatically rearrange for best fit and graphics will automatically resize and get smaller on smaller screens. This is important because it allows your process automation software to work on the devices that your people are using – any of them.

Business Rules / Logic

Logic is the biggest challenge to rapid development. As mentioned above, common types of logic are often built into the development environment as parameters. What this means is that if you are essentially performing basic CRUD operations on your data, you may not need any custom coding at all. Tools that promise "no coding" often rely on the fact that most applications don't do any really fancy data manipulation.

To best understand business rules in a development context, let's consider an example application and break it down. Imagine you are automating the expense request process. You might design a screen that looks something like the one below.



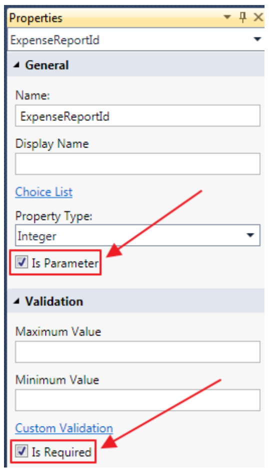
Expense Request Form

Requested By	Bob Smith Carrie Jones Derek Williams Eddie Gomez
Date Needed	11-28-2013
Description	50' roll of plastic sheeting.
Est. Cost	$14.95
Department	Accounting Business Development Customer Service Field Service
Approve Date	
Approved By	

To make this form work, each input must be validated. Let's see if this can be done without coding:

Requested By – this needs to be tied to a list of valid users. The users are probably stored in another list, so that list needs to be retrieved and provided as valid options for this field. This is a common function in many development tools and does not typically require coding. You may also want to default this value to the current user. Many tools allow you to get system values like the current user's login name by specifying a parameter (or function as it is commonly called).

Date Needed – most tools have special date/time input objects in their UI builder. These objects come with a set of built-in parameters for handling dates and ensuring they are valid. However, you may have additional constraints. For example, you may need to allow at least three days for processing a request, so this input cannot be less than three days from now. This type of contraint is also commonly done without coding by specifying a value that looks something like: ***>=[Today]+3*** inside of the "minimum value" parameter. You will also likely make this a required field, which is yet another parameter. The screenshot on the right shows how parameters may be represented on the screen as the UI is being built.



Properties
ExpenseReportId
General
Name: ExpenseReportId
Display Name
Choice List
Property Type: Integer
☑ Is Parameter
Validation
Maximum Value
Minimum Value
Custom Validation
☑ Is Required

Est. Cost – there is probably little validation that goes into this input other than to make certain it is not zero. You might have a minimum value parameter here, too and this is also required data. However, this data will likely be evaluated to determine who the approver needs to be. It is often the case that the greater the expense, the higher level the manager that must approve it.

Department – the Est. Cost may be combined with the Department input to derive the correct person to approve the request. The two values essentially provide input for a lookup in an Authorized Approver list to find a username. What happens next is called an **Event**. Events (or Event Triggers) are built into BPMS systems, but they are also part of many other rapid development tools. These tools have objects, similar to the drag & drop objects in the UI builder, which perform messaging functions (sending emails, texts, or even screen pop-ups). This allows for alerting the Approver that they have a task to perform. The system could even provide a link to the proper screen.

Rapid Data Deployment

One of the newest tools for using data elements within an application is called **Reflection**. This generally only works when the data store is a relational database. Because this type of data storage relies on predefined relationships of the data, a lot of information about the layout of the UI can be extrapolated from it. Systems like Microsoft's Lightswitch can create a default screen that is often close to what you might have built yourself. This allows you to leverage the work you put into designing how the data will be organized into other parts of the application.

Choosing a Development Ecosystem

Armed with a basic understanding of what development tools can do, you are ready to start deciding upon a specific tool. The first thing you should do is choose from the two broad categories of environments.

Open Source

The only thing that you can be assured of with an Open Source product is that you will have access to the actual programming code used to build the tool. This doesn't do you much good unless you decide you want to get involved in enhancing the development environment you choose. Chances are pretty good that this is not your thing. However, many Open Source products are distributed for free, and that makes them attractive. Some are also supported by large user communities that provide Q&A in free online forums.

Not all Open Source products are free nor do they necessarily come with any free support. Most of them are based on Open Standards, which means that there will be a large number of professionals who can assist with their use. If you buy an Open Source product from a commercial enterprise, you will probably not notice much difference between this and the next broad option.

Proprietary Solutions

You will not get to look at the source code of a proprietary solution. However, you may get better documentation and better support. You will also quite likely pay more for it. There are some very big names in the proprietary software development tools marketplace. This is a good thing because it is very risky to develop software on a platform that disappears out from under you later.

Blue Chips

As mentioned, there are some big names in the software development tools business, but not all of them are in one camp. For example, Google and IBM, two very large players, both offer solutions that are open source. Microsoft and Oracle, two other big players offer proprietary solutions.
Even these blue chip players may not be able to offer you everything you need. Some products can be added into the core development environment to add more capabilities.

For example Dundas offers extensions to several development environments that can provide a rich business intelligence capability with limited effort. There are many such add-ons for UI controls, as well as business logic and data layer functionality. Mixing and matching these add-ons can help you create an environment that has the features you need to complete your project even more rapidly.

Choices, Choices and More Choices

Even within Microsoft's ecosystem, there are three possible choices for RAD:

1. **Access** – this is a traditional option, but may be handy if you already have Office Pro installed on all your computers as it is included. Beware – this

product does not work well for more than two or three simultaneous users and if you are going to have large amounts of data, you need to use SQL as your data store and just use Access as your frontend.

2. **SharePoint** – for web portal type solutions, this is a good option. It has limited functionality out of the box, but with the paid versions you get more BI and customization capabilities and there are many add-ons to enhance its functionality.
3. **Visual Studio / Lightswitch** – this is the newest edition to Microsoft's RAD line and it has many of the modern capabilities you might expect including reflection and a complete HTML/JavaScript interface (be advised that it can also use SilverLight as its interface which is very proprietary). If your application is primarily CRUD based, you may not need to write any code, but if you do you will have the choice of Visual Basic of C# (pronounced see sharp) for any coding you may need.

The most popular Open Source environments are PHP, Java and Ruby on Rails. There is a large ecosystem of products available around the PHP and Java programming languages to accelerate development. Eclipse is the most popular framework for PHP and Java programmers. The Google Web Toolkit (GWT) has some RAD capabilities. Ruby on Rails has its own framework, similar to Microsoft's Visual Studio / .NET environment that has an integrated development environment (IDE) as part of the basic offering. There are also many add-ons for Rails.

IBM, SAP and Oracle all have development environments. It probably makes little sense to use any of these toolkits unless you are already invested in their environments. Of course, Oracle purchased Sun Microsystems, the developer of Java, so this common development language is now an Oracle product, but there are third party environments for developing in Java if that is your language of choice.

As mentioned, BPTrends has written extensively about BPMS software, so I will not offer details about these tools here, but for many types of process automation, they can be highly effective and have the advantage of allowing you to start building a central repository of process automation systems. Most of them also offer better tools for review and analysis of your processes as you continue to enhance them.

Finally, there are all the others. This would be a much longer document than intended if I were to review all of the many RAD tools out there. Wikipedia has a nice list if you would like to do your own research (Wikipedia RAD Tool List).

Hopefully, this overview will provide you some perspective that will assist you in making the right choice should you decide to go it alone in building process automation from scratch. Choosing software when you know what you want can be a challenge. If you are new to software development, this one is a very great challenge indeed. Take your time and be sure to involve anyone who may be using the tools in your selection process.