Why Programming Language Matters

by Mark Lipsitt

When considering the purchase of a table, golf cart or chair for a club, how the item was made and with what materials are important purchase considerations. Is it ash or oak or cherry? Is the seat cover vinyl or made from a natural fabric? How are the legs attached? In other words, is it well constructed and what was used to make it that way? These questions matter because they tell you whether the item is going to last and what you can do with it.

When clubs begin the process of selecting software, the issues that cause them the most pain are support responsiveness, functionality, whether the program is user-friendly and what the training will encompass. All of these are perfectly appropriate things on which to focus, but there is another key element that is, unfortunately, most often excluded: the programming language and technologies (the materials) used to build the software. The materials matter just as much in this situation as with any other purchase.

It is somewhat understandable that most clubs don't consider the specific programming tools used by the vendor in creating its programs, or the impact those tools might have on the future use of the club. Most clubs don't have a significant information technology presence to help them with these issues, certainly no one dedicated to that function. However understandable it might be, it can still be a problem.

Lack of understanding of this issue and the programming tools in question has lead to a proliferation of poor choices for the club population at large. That's not to say one must be an expert, or be able to debate the merits of one programming language vs. another. In fact, if you put 10 techies in a room, you would probably have 12 different opinions as to what are the best tools for a particular situation. However, there are tools and methodologies that most would agree are clearly wrong.

Here are examples of methodologies to avoid.

1. One of the leading software programs in the club niche is not even built on a relational database, but uses a flat-file structure instead. This is widely considered to be poor design and can contribute to significant inefficiencies in processing and reporting information.

2. One of the leading software programs in the club niche is built using a programming language that many technical people have never heard of. This can be a significant problem in trying to integrate other applications and if the company ever went out of business you would be hard-pressed to find anyone to support the application.

3. One of the leading software programs in the club niche is built on a database platform that is not designed to handle more than a relatively small number of users at one time. Obviously in a club of any size and complexity this will be a problem.

These are just a few examples of undesireable programming languages, databases and tools being used. And what is worse is that the provider companies are significant players. The potential impacts have been alluded to, but let me delineate further the reasons these choices matter. Using the wrong technical tools can have one or many of the following results:

- Significant reduction in the speed of processing, either in processing a transaction (such as POS or posting) or in generating a large report;
- Issues of user and record locking within the database, in other words, users can experience getting locked out of modules or system locks when too many people are processing transactions;
- Data integrity issues, with frequent corruption and loss;
- Extreme difficulty in integrating with other software applications, sometimes an inability to do so at all; and
- Inability to find knowledgeable assistance should your software provider ever go out of business.

It is always important to consider things like support, the partnering relationship, functionality, implementation issues and training. But it is equally important to consider the tools used to build an application and the database platform upon which the application sits. Ask for a couple hours of time from a local technical firm. It won't cost much to do so and could cost you in many ways if you don't.

A well-built chair should not only be the right size and shape, the right color and fabric but also needs to have been well put together, using the proper methodologies, tools and materials. Software applications are no different. To make the best possible choice and to avoid unwanted problems down the road include tools, materials and methodologies as part of your software selection process.

Mark Lipsitt is CEO of Northstar Technologies and formerly principal and founder of The Lipsitt Group. He can be reached at (720) 851-8373 or mark@globalnorthstar.com.