DB/C Newsletter

January 2006

News and Comment

There continue to be many users of the DB/C programming language (and of DATABUS in general including our competitors). Outside observers would probably wonder why a 30 year old language continues to be used to develop mission critical business-oriented applications. There are several reasons it continues to prosper. One of those reasons has to do with fundamental technical design decisions made by the original designers of the language. This month's article deals with some of those concepts.

Disclaimer - as it's an opinion piece, this month's article has my byline. I'm not an expert in Ruby or Rails. The information in the article is the result a few weeks of my learning about Ruby and Rails.

don.wills@dbcsoftware.com

What About Ruby?

by Don Wills

In my opinion, Ruby is not a major advancement in the world of programming languages. It has some interesting concepts, but I believe when the hype dies down, it will become just another programming language in the PHP/Pearl/Python family of scripting languages.

The hype generally goes something like this - using Rails, you can create typical web-based CRUD (create/retrieve/update/delete) data base maintenance programs in one tenth the time when compared with X (name your own X). Another commonly hyped point is that it takes many fewer lines of code and/or keystrokes to express the same program when compared with other languages. Actually, these two claims are generally true, but they aren't really important to most commercial software developers.

Let's examine the first claim, that using Ruby on Rails, you can create typical CRUD programs quickly. Yes, but that's a feature of Rails, not Ruby. It is true that Rails uses some neat Ruby features to do its work, but a tool like Rails can be built for many programming languages. Rails is just another snazzy program generator for building web apps. Besides, today most professional developers using other languages are not building CRUD programs from scratch - they are using program generators or other automated tools to do the grunt work for them.

One of the best features of Rails is Active Record. It is yet another SQL based object-relational mapping (ORM) mechanism. Active Record is different than other ORMs in that it is generally stateless. This means that connection pooling and other performance improvement methods do not impact the programmer's use or design of programs. That is a good thing. But as noted above, there is nothing unique about Rails and Active Record - Active Record can be (and is being) developed for several other programming languages.

There are several other reasons why Ruby isn't this next big thing. Here are a few -

Performance is an issue - execution of Ruby is interpreted from the abstract syntax tree of its source. Compilation simply doesn't happen, either to bytecodes or to native executable code.

Ruby, like Python, Pearl and PHP, is actually older than Java. These languages have had plenty of time to gain traction, but they have not attained the status of younger languages like Java and C#.

Ruby has been lauded as simple and clean. In my opinion, that's just not true. In designing new things, I've always followed the Principle of Least Astonishment. That means that one can generally guess what an operator, verb or syntactic element does just by looking at it. Ruby fails in this regard. It is true that Ruby is cleaner than the other P languages, but that's not saying much.

Ruby isn't useful for server-side or desktop (GUI) programming.

Ruby doesn't support fixed decimal arithmetic or Unicode character values.

Ruby is dynamically typed. Some people like this; I don't. A dynamically typed language allows a variable's type to be changed at will. At one point the variable may be an integer type, and just by assigning a string to it, it changes to a string type. Errors caused by invalid use of dynamic typing cannot be found by the compiler. They are found at run time (hopefully during testing!).

Ruby has poor IDE support. One of the advantages hyped for Ruby is that you need to type fewer keystrokes. That's true, but with Eclipse's Java code completion or Visual Studio's Intellisense, that's irrelevant.

As a student of programming languages, I'm always interested in learning new languages. Ruby has some useful features that will be copied in other programming languages. But Ruby will not be the next big thing in the world of programming languages.
