**The Education of a Programmer: A Review of Job Types and Academic Degrees**

Matt Weisfeld

* By [Matt Weisfeld](http://www.informit.com/authors/bio.aspx?a=48da22e3-dc47-463a-9d15-b68bad0bd64b)
* Oct 4, 2013
* Print [Print](http://www.informit.com/articles/printerfriendly.aspx?p=2140743)
* Share This[Share This](http://www.addthis.com/bookmark.php)
* Discuss[Discuss](http://www.informit.com/articles/article.aspx?p=2140743&WT.mc_id=IT_NL_Content_2013_10_10#articleDiscussion)

**Page 1** of 1

**From the author of**

[Object-Oriented Thought Process, The, 4th Edition](http://www.informit.com/store/object-oriented-thought-process-9780321861276?w_ptgrevartcl=The+Education+of+a+Programmer%3a+A+Review++of+Job+Types+and+Academic+Degrees+_2140743)   
  
[Learn More](http://www.informit.com/store/object-oriented-thought-process-9780321861276?w_ptgrevartcl=The+Education+of+a+Programmer%3a+A+Review++of+Job+Types+and+Academic+Degrees+_2140743) [Buy](http://www.informit.com/buy.aspx?isbn=9780321861276&w_ptgrevartcl=The+Education+of+a+Programmer%3a+A+Review++of+Job+Types+and+Academic+Degrees+_2140743)

**Professor and software engineer Matt Weisfeld explores the following two questions: first, what categories of programming careers exist and, second, what degrees/certificates are available in pursuit of these programming jobs?**

*It's the only job I can think of where I get to be both an engineer and an artist.  There's an incredible, rigorous, technical element to it, which I like because you have to do very precise thinking.  On the other hand, it has a wildly creative side where the boundaries of imagination are the only real limitation.  ~Andy Hertzfeld, about programming*[*http://www.quotegarden.com/programming.html*](http://www.quotegarden.com/programming.html)

**Introduction**

As a college professor, I see a lot of students who are considering educational opportunities in programming.  While some are freshly minted high school graduates, quite a few are people who graduated several years ago (from both college and high school) and are seriously considering a career change.

Before I started teaching, I held various positions in the software development industry for about 20 years.  With experience in both industry and teaching under my belt, I have spent the past several years advising students on how to begin their programming education in their quest to become a software development professional.

In this article, I explore the following two questions: first, what categories of programming careers exist and, second,  what degrees/certificates are available in pursuit of these programming jobs?

**Programming Jobs in the IT Industry**

Recently, a board of Information Technology (IT) executives in Cleveland convened with the express purpose of defining what types of IT skills they needed the local educational institutions to focus on. The goal was to work closely with these schools so that graduating students would have the skills required by these companies.

These executives identified two primary categories of IT jobs: *producers and consumers*.

* **Producers** of IT *create* IT products and services.  Examples of producers are Apple, Microsoft and Hyland Software. IT products and services are their primary business, which they sell to individuals and other businesses. These companies are often looking for programmers whose strength is in the analytical process of creating and maintaining systems from scratch.
* **Consumers** of IT *use* IT products and services in their daily business operations.  Examples of consumers are Huntington Bank, Eaton and Sherwin-Williams. These companies use IT products and services to implement their business systems and solve their business problems. Their needs require more business process knowledge, portfolio management, and soft skills.

While quite a few of the lines between *producers* and *consumers* may be blurred, meaning that some companies that are considered primarily *producers* will also be *consumers (and vice-versa)*, I believe that the model is quite sound.

For example, while Microsoft’s focus is to produce products for sale, they are also a business and must perform HR, payroll and other business functions. Thus, they may well, and almost certainly do, consume IT products and services from other companies. Nonetheless, Microsoft main purpose is to create IT products and services– thus they would be classified as a *producer*.

On the other hand, the vast number of IT employees at Huntington Bank consume IT products and services, but don’t actually create them. Their functions would be to integrate, enhance and maintain their systems (primarily purchased from other companies), not necessarily create IT products and services from scratch. That said, many companies that are considered consumers, such as Progressive Insurance, also produce new code, especially for applications such as websites and mobiles apps.

An important point to note in this discussion is the use of the term Information Technology (IT). Conceptually, the *producer* and *consumer* model falls under the umbrella of IT; however, the definition of IT itself can be blurred as well. For example, there are many categories that can be a subset of IT and/or overlap with IT: Computer Science, Business Analysis, Software Development, Software Engineering, among others. Since this article is focusing on programming, where does programming fit into this model?

Generally (although certainly not exclusively) programmers who are *producers* fit the model of the traditional programmer - those who sit at their computer and code, happily creating products and services.  And, while programmers who are *consumers* certainly do write code, they are primarily configuring and supporting products written by other companies. For example, writing code to program appliances in your smart home is a *consumer* coding activity while the *producers* are the programmers who created the smart home system.

With the two basic categories of programmers defined, another relevant question is: which category accounts for the majority of the programming jobs? If we were asking this question 25 years ago, the answer might well have been *producer*.  In the 1980s, most companies developed their own systems such as payroll and expense reimbursement. Thus, there were a lot of programmers actually *producing* code in the sense that they were writing code from scratch and maintaining legacy systems that were developed in-house.

During the tech boom of the 1980s and 1990s, tech companies such as Oracle and SAP started producing specialized software products to handle many of these in-house). Today, not many companies actually write their own payroll or expense reimbursement software from scratch. -They purchase software products and/or services to handle these types of functions.

As a result, in today's business environment, much of the business programming does not require writing much in-house software; however, it does require a lot of configuring, enhancing  and implementing third party software that includes various levels of coding. So while there are still many programmers that write products and services for sale (*producers*), there are far more programmers who are involved in maintaining business systems with third party software products and services (*consumers*). This is, no doubt, due to the fact that just about every company in existence depends on programmers to operate their company’s business systems.

Now that we have defined the categories of *consumers* and *producers*, we can explore the second of our questions : what type of educational options are available in programming?

**College Degree Tracks**

Programming skills apply to so many modern professions, that it is often difficult to really nail down just a few applicable degree categories. Nonetheless, we have to start somewhere, so I decided to look at various academic institutions to see how they approach this question. Generally, if you want to pursue a degree in programming, there are three types of programs available: Information Technology, Software Engineeringand Computer Science.

Despite the fact that there are three categories listed, it is very important to understand that good programmers can, and often do, move freely amongst the three groups.

Regardless of the categories, all programmers must become very familiar with the software life cycle. Whether you are writing products or supporting business systems, you must be proficient with writing code. This means understanding proper analysis, requirements, implementation, testing, and quality assurance.

Generally, in this *consumer/producer* model, I would assign the Information Technology degree track to the *consumer* category while the Software Engineer and Computer Science tracks would fall under the *producer* category. Although, I reiterate that professional programmers often move freely amongst all of them.

**Information Technology**

In many academic tracks, programming business systems would fall under an Information Technology degree (IT). Virtually every organization ( corporate, academic, non-profit) has a function called the IT department. These departments are tasked with maintaining the business systems of the organization. For example, the IT department normally maintains computer hardware, networks and software applications.

The IT department supports the business infrastructure of the organization. The cool thing about supporting a business infrastructure is that it spans the entire organization. There are IT people underlying virtually every aspect of the organization – products and services (parts, labor, inventory), business systems (vendors, customers, accounting) and internal support (human resources).

In many academic settings, the Information Technology degree falls under the business school. In most MBA programs, there is a concentration offered in Information Technology. In fact, if a student is looking to make a career change and is considering more education to prepare for a programming degree, I make sure they consider an MBA (with a concentration in IT). In the long run, a graduate degree may prove more useful than a second undergraduate degree.

Finally, there are many variations to the IT theme. I have seen degrees that are virtually identical to Information Technology, yet are called something like Management Information Systems (MIS), Computer Information Systems (CIS), Information Systems (IS) and a few more that I can’t even remember. The bottom line is that you have to pay special attention to the actual curriculum because you can’t always tell by the name of the degree program.

**Software Engineering**

Based on my experience, software engineers create software that is part of a system developed by a manufacturing/engineering firm. While IT professionals often write software for internal support functions, software engineers generally write software for industrial and consumer products (sometimes called shrink-wrapped software). Software engineers often work hand-in-hand with other types of engineers such as electrical engineers and mechanical engineers.

For example, suppose that a company produces automation products that control assembly lines in a factory.  These controllers are hardware products that are designed by electrical engineers and mechanical engineers. In today’s marketplace, just about every type of hardware product is run by software or embedded firmware. Software engineers produce this software/firmware.

Often, an employee with a Software Engineering title will be programming directly onto computer chips rather than a general purpose computer. Examples would include uploading software to cell phones, gaming platforms, automobiles and actual computers. While there may be exceptions to this, I have never come across a business programmer (*consumer*) who uploads code to a chip.

**Computer Science**

Despite the fact that I have worked at various jobs covering Information Technology, Software Engineering and Computer Science, when someone asks me what my profession is, I almost always answer – Computer Science.

However, I often get into pretty interesting arguments with software development professionals about how to define Computer Science - and the opinions are quite varied.  More to the point, Computer Science is often about machine-level activities such as writing algorithms, enhancing hardware and software performance, writing operating systems and writing compilers. Computer scientists may also spend a lot of time working on enhancing the performance of a wide variety of things such as data structures, networks, run time environments, and memory.

I have worked for three software startups that were considered Software Engineering firms. However, the work we did was based largely on Computer Science. One firm produced simulation tools, one database mapping tools and the other performance tools. In each case, the tools were often sold to consumer companies for their IT departments. This is just a great example of how three of these categories, Information Technology, Software Engineer and Computer Science, while distinctly different disciplines, can, and often do, intersect.

**Conclusion**

In conclusion, I can't stress enough that these categories, from both the industry and the academic side, are meant as guidelines. These guidelines are a great resource in advising students on what types of programming jobs and education options are available. While they are obviously not cast in stone, I believe that they offer a very strong position to start the discussion on what it means to be a programmer.