PROFESSIONAL COLDFUSION CONTENT FOR THE COLDFUSION PROFESSIONAL

The Fusion Authority Quarterly Update

Vol. II Issue I

SPECIAL BUSINESS ISSUE:

Business practices and concerns to channel your programming skills into cash

- How to Avoid Unpaid Consulting
- Making Google Pay
- Deconstructing the Consulting Contract
- Plus — Developing Applications with Tranfer

And Much, Much, More...

VOL.II ISSUE I $14.95

ISSN 1932-0264

9 771932 026000 21>

CONTENTS

Fusion Authority Quarterly ■ Vol. II ■ Issue I

Editorial

Columns

Concepts

Special Business Section

Business Management

Team Management

Personal Growth

Tools

Riding the Coming Wave

by Hal Helms

In his 1841 book, *Extraordinary Popular Delusions and the Madness of Crowds*, Charles MacKay catalogs numerous examples of speculative bubbles, detailing their unlikely rises and spectacular falls. Anyone who survived the internet bubble of the 1990's may find some comfort in knowing that the internet craze, in which all things internet became so enormously overpriced that the only place for prices to go was down (and down they went in spectacular fashion), is but the latest in a long line of boom and bust cycles. Take, for example, one of the unlikeliest of bubbles MacKay outlines—the "tulipmania" that overwhelmed Holland during the 1630's. During that time, the popularity of trading in tulip bulbs spread from the upper to the middle and then to the lower classes. It seemed that everyone knew that tulip bulbs were a sure thing and huge fortunes were spent to acquire the rarest and most prized of these. People sold all they had to "invest" in such a once-in-a-lifetime opportunity.

And then came the bust. Bulbs were devalued by 90%--and even then, there were few buyers. There are few sure things in this life, but one of the surest is the repetitive nature of the boom and bust that defines a bubble.

So, let's run time backwards to 1999. The dot com boom is cresting and the need for internet techs is so great that anyone that can spell "HTML" has multiple job offers. Technically skilled people are hounded by headhunters with offers of stock options and signing bonuses. For many of us, this is a golden age. With a smattering of CFML, HTML, JavaScript, and SQL (with perhaps a bit of CSS thrown in for good measure), we have all we need to participate in the internet boom. Life is good.

Now it is the year 2000... and the bust is here. Just as the Dutch discovered, "investors"—speculators, really—in risky, ill-thought internet ventures find that what goes up must come down.

Many of us suffered from the aftermath of the bust: salaries became stagnant; benefits were diminished; offers from headhunters disappeared. Now, having lain dormant for years, the web is generating new interest. This, too, is to be expected; unlike tulip bulbs, the internet represents a genuine innovation—something economists refer to as a "disruptive technology". But where does that leave us as internet developers?

During the internet bubble, the great demand was for people who could create websites with, perhaps, an e-commerce module bolted on. Over time, those simplistic demands gave way to the need for more full-bodied web applications. Many web developers, seeing this trend, have worked to upgrade their technical skills. In the ColdFusion world, consider the profusion of frameworks and the adoption of object-orientation as emblematic of this recognition that things are different.

But something new has emerged during the period between the discovery of the internet and the present that makes such seemingly fine efforts misguided. That "something" is outsourcing and, depending on your point of view, it either promises or threatens to change the face of web development and our role as web developers.

Meet Nick. Nick is a 27-year old web developer. He has nine years of web development experience. He writes equally well in SQL, ASP.NET, CSS, XHTML, and JavaScript. He incorporates AJAX into his applications and "gets" Web 2.0. Nick is well positioned to benefit from the rediscovery of the web. You probably know someone like Nick or you may well have equal or greater qualifications. But there is one difference: "Nick", you see, is (pardon the pun) a nickname; he uses it because his clients are more comfortable with it. His real name is Nitish and he lives in Hyderabad, India. Nitish has a number of American clients who value him for the high-quality work he produces. Nitish's clients appreciate that he has a network of similarly-skilled developers available for larger jobs. Although you've never met him, Nitish is your competition.

Are you an expert in web development? So is Nitish. Are you upgrading your skill set? So is Nitish. Are you keeping abreast of the latest technologies? So is Nitish. While you may have much in common with him, there is one area that Nitish stands apart: Nitish feels himself very well-compensated at $14.00/hr. Nitish is eating your lunch.

Most web developers in so-called "first world" countries ignore Nitish. But that's beginning to look less like strategy and more like denial. There are just too many "Nitishes" to ignore. Already, larger companies want Nitish—and all his friends and colleagues with him. Many of the issues that cast a pall over the early enthusiasm outsourcing produced, such as language and culture barriers, time difference, and physical distance, have been removed or, at least, largely mitigated. What's left is less and less reason to pay programmers like you or me three or five or ten times as much as Nitish wants for doing the same job.

There is some good news: while outsourcing is inescapably attractive to large companies, the small-to-medium businesses don't know Nitish. But their ignorance is more respite than refuge. Economists have a term for the condition where one product costs substantially more in one location than another. Such markets are said to be "inefficient". The bad news is that markets become more efficient over time. And companies similar to the one Nitish works for are working hard to make that particular market become as efficient as possible.

What does all this mean to us as web developers? There can be no possible difference in skill or productivity sufficient to justify the cost disparity between the first and the third-world developer. More of the same—in the sense of acquiring greater technical proficiency—can, at best, merely slow the inevitable.

In this issue of the *Fusion Authority Quarterly Update*, much has been written about the non-technical aspects of being a software developer. As developers, as technicians, our natural tendency is to immerse ourselves deeper into the technical aspects of our profession. This is important, of course, but I believe our actions on the non-technical—business —aspects of our profession over the next several years will be absolutely critical in determining our success as developers, whether we choose to do that as independent consultants or as employees for a company.

In 1982, John Naisbitt wrote the book, *Megatrends*. In the book, he identifies ten socio-economic trends he thought likely to change the landscape of life and business for many people. With hindsight, we can recognize how accurate his analysis was. Megatrends are to normal trends what tsunamis are to normal waves. Megatrends (like tsunamis) develop slowly, inexorably. With both tsunamis and megatrends, the tide that breaks with such profound effects began its life with little fanfare. For that reason, both tsunamis and megatrends are understandably hard to spot.

Yet one might expect experts in their fields to identify megatrends more readily. Sadly, they are not. Experts are invested in the current system--a system that megatrends often threaten. The natural self-interest of experts is often identified with the continuance of the current order and bold thinking is seldom rewarded by the keepers of the current model.

What is true for "those" experts is also true for us. The emergence of the outsourcing of development is a megatrend in the software development world—one that we must understand the importance of—or risk being swept away as the wave of outsourcing breaks upon us.

For as long as most of us have been alive, software development has been predominately a phenomenon of the local economy. If you were an employee, you went to work for a local company. If you were a local company, you tended to solicit and find work in your local area.

Nitish and his colleagues did not grow up in this environment. Their "local economy" is the developed world and they're offering the same product you are—only at a greatly reduced rate. The continuing refinement of communication technology can only make their offerings more attractive.

I believe there is a strategy for making outsourcing work for us—to be a promise rather than a threat—but it does involve us rethinking our roles as developers. Outsourcing is beginning the process of turning technical programming into a commodity. Commodity markets are typically fiercely price-competitive,

which is why Nitish and his friends are enjoying the present situation. What can we do? Listen to what Fred Brooks wrote in 1987 in an article titled "No Silver Bullet":

> Not only are there no silver bullets now in view, the very nature of software makes it unlikely that there will be any—no inventions that will do for software productivity, reliability, and simplicity what electronics, transistors, and large-scale integration did for computer hardware.... I *believe the hard part of building software to be the specification, design, and testing of this conceptual construct, not the labor of representing it and testing the fidelity of the representation*.... If this is true, building software will always be hard. There is inherently no silver bullet..

What does it mean to be a developer? I think most programmers who would answer this question would concentrate on languages, data structures—perhaps design patterns. But Brooks says that the hard part of building software is the specification, design, and testing, not the actual implementation. If Brooks is right, this points us to a different concept of "developer". In a world in which Nitish et al. provide a cheap programming resource, we can find our niche by moving up the food chain. We can concentrate on the design and architecture of the application and outsource the fabrication of what we design. Now, instead of being a threat to us, the rise of outsourcing can be a strategic advantage we benefit from.

But what of Nitish? Can he not acquire design and business skills — just as he learned technical skills? The nature of design is such that it does not support the outsourcing model; it relies on a shared vocabulary and shared concerns. The designer must internalize the nature of the problem so that he can then craft a solution. Design is a high-touch approach to building high-tech products. For custom software, there is simply no substitute for being there.

For some, this argument will fail to convince. They will rely on their supposedly superior skills to bridge the gap in costs between themselves and Nitish. They may argue that programming—"real programming"—is an art that can never be commoditized. This romantic view is understandable: the people whose business models are threatened by a new paradigm are the ones least likely to see the megatrends that are reshaping the landscape. Unfortunately for these romantics, they are not the ones who will ultimately weigh in on the question and their arguments are unlikely to hold sway over the jury of businesses that must decide on one of two models of local vs. outsourced programming. We are not required to like it, but we ignore outsourcing and the coming commoditization of programming at our own peril.

Hal Helms writes, speaks, and consults on software development. His podcasts with Jeff Peters can be downloaded from http://helmsandpeters.com. His popular "Occasional Newsletter" is available at http://halhelms.com. Hal can be reached at hal@halhelms.com.