BPTrends COLUMN

Harmon on BPM September 2021

Improving Productivity

The last 50 years have witnessed a strange phenomenon: the quantum improvement in information systems productivity. Leading economists and IT optimists have constantly predicted that investment in information systems would lead to major improvements in overall productivity. Other economists and IT pessimists have consistently undertaken studies that suggest that IT is generating little or no increase in overall productivity. In fact, major IT-driven efforts have been made to improve productivity and seemingly failed, for a decade or more, and then suddenly succeeded. It's as if some huge resistance had to be overcome, and finally was, but only after a decade of failure. Consider three cases.

Integrating the Flow of Business Process Information

There was a huge interest in business process improvement in the early Eighties, inspired by books by Michael Hammer (Business Process Reengineering) and Bill Davenport. For awhile there were conferences, reports of major BPE projects and then stories of failures. It seemed as if Hammer and others had oversold the idea that IT could revolutionize business process work, and after some major failures, companies began to look elsewhere for improvement gains. In essence, Hammer reported on projects that relied on the development of proprietary communication systems. Using such proprietary technology, one could link up multiple software systems and pass information, allowing a whole new level of coordination. In fact, however, linking such systems was very hard. Different interfaces, different protocols, a lack of developers with all the necessary skills, and so on, assured that large systems that coordinated lots of distributed applications would be very hard to develop and even harder to maintain. Very large organizations with large teams of software developers could and did pull it off. Most organizations lacked the resources and launched projects that were quickly overwhelmed with difficulities.

By the late Nineties BPR was largely winding down and IT departments began to focus on Y2K problems, and on off-the-shelf software (e.g. SAP) which solved integration and communication problems by all being developed by a single company with a single protocol.

As interest in BPR was dying, however, another technology became popular – the Internet. The Internet had been around for awhile. It had originally been developed in the Sixties, by DARPA, to assure that communications would continue to occur in the event of a nuclear war or a similar disaster. It relied on government controlled, standard protocols that used the existing phone lines. Messages were sent as small packages and could follow various routes to assure they got from point A to point B. For awhile the Internet was resisted and only used in universities. No leading technology company wanted to embrace a non-proprietary standard that they couldn't control.

The invention of the World Wide Web at CERN (the European Nuclear Research Consortium) changed all that. Suddenly there was this graphic, online world that could be transversed with ease. With the Web came easy to use web browsers and suddenly email was popular. In spite of a decade of resistance, both ordinary users and companies abandoned their various proprietary communication standards and embraced public, open standards. And suddenly a new realm of process improvement, based on Internet standards like XML, HTML and IIOP came into being. Suddenly one could develop process models and specify connections between existing activities and connections could be easily made. Suddenly the world that Hammer and Davenport described in the Eighties became possible.

The BPM Software tools that become popular in the early years of this Millennium are simply tools that create connected processes by utilizing Internet protocols. The 80s dream of building super highways became the reality of the first two decades of the new millennia.

Object-oriented Programming

Now consider the adaption of object-oriented (OO) programming techniques. Object techniques were invented by computer scientists in the early seventies and were used at Xerox PARC to create the Smalltalk language and early user-friendly interfaces. Artificial Intelligence (AI) developers used OO techniques in several expert system tools to capture networks of human knowledge, and Steve Jobs famously discovered Smalltalk at Xerox PARC and borrowed it's concepts to create Apple's Lisa, and then the Mac computer. I was involved in both Expert System and Apple efforts to introduce OO

techniques to programmers who were trying to develop apps for Macs and for expert system applications. Later, in the early 1990s, I wrote the *Object Strategies Newsletter* and promoted the use of object databases, object oriented software tools, like ObjectVision, and languages like C++ and Object Pascal. The Object Management Group (OMG) was formed to promote object technologies as a way of linking various applications.

Everyone who studied the field ended up agreeing that programmers would be a lot more productive if they adopted object techniques. By the Nineties, programming projects were becoming distributed and incredibly complex, and object technologies modularized the development effort and facilitated the use of modern software development methodologies. In spite of that, with only a few notable exceptions, object technology wasn't widely adopted, and the predicted productivity improvement weren't achieved.

Things changed, suddenly, in the middle of the Nineties. The World Wide Web was launched in 1989, grew slowly for a few years, and then took off with the introduction of powerful web browsers like Mosaic/Netscape and then Microsoft's Internet Explorer. Once interest in web browsing became hot, developers rushed to learn about how one developed web applications. Attention turned to Sun's Java language and then to its various successors and the whole online world blossomed.

What's important for the purposes of this article is that Java was an object-oriented language, a natural heir of Smalltalk and C++. No one seemed to care that Java was on object oriented language – Java was **the** Internet programming language. With some limitations, Sun provided Java as an open standard. Programmers didn't shift to an OO language, they shifted to an Internet/Web language, and the rest is history.

Remote Work Facilitated by the Internet

Now consider a recent transition. There have been experiments, on and off, for thirty years, on work from home. With the invention of the computer and the adoption of the Internet and the Web, the idea of employees working at home, on computers had become an obvious possibility. But the actual practice has been very tentative. Somehow there was never any pressing reason to give it a serious try. Then, overnight, a worldwide COVID-19 pandemic changed everything.

Although not much has been made of it yet, we predict that, in hindsight, many will note that the world of business, in the US

transitioned to home-work in 2020 and continued almost without a hitch. Obviously businesses like travel, restaurants and hotels – businesses that depended on the actual presence of customers – closed down. But most businesses – businesses that moved food to stores and provided services of all kinds – largely continued without any real pause. Luckily for all, the pandemic occurred in the 2020s, when customers already had computers and were used to email and online Web-based shopping. Amazon already existed and its processes were already well established; Amazon simply went into high gear. Similarly, most employees already had computers and modem links at home, and were able to simply sign-on and enter their corporate computer environments with little difficulty. Employees shifted and most office work proceeded during the pandemic without a major disruption.

Now we will see what happens as the pandemic fades. Some will return to work. Some will stay at home, and many will adopt a mixed pattern that includes more home-work than in the past.

The point of all this discussion is that technological transition does not occur smoothly. First there needs to be a technology base. Sometimes transitions are attempted when the base is inadequate and they fail. There must also be a clear vision, and there must be supportive technologies in place to support the transition. There must also be a real driver, something that excites people and causes people to give the new a try. Resistance to change is strong; it takes a real wave of enthusiasm, or a real crisis to force people to try a major alternative.

Technology change takes place in quantum steps: No change for awhile, no increase in employee productivity, and then suddenly quick change and a jump in productivity. As the examples indicate, groups can gather together, be inspired by a vision, work to implement change, and make little progress. Then, when it seems as if change is to be delayed forever, that change takes place rapidly and involves many people who were never involved in the buildup effort.

It makes one wonder what transition is building up now. Lots of people were laid off during the pandemic. In some cases they are being rehired. In other cases the employees were already redundant, but hadn't been fired to avoid labor problems. Now, having been fired "because of the pandemic," management will not rehire them to improve productivity. Similarly, faced with labor shortages now, other businesses will introduce new technologies as an alternative to fewer employees. We have already read of phone ordering in restaurants to

make up for a lack of waiters. Changes in how offices are organized and how customers interact with companies will continue, surely, but what else?

Using technology to improve productivity will continue. New technologies will struggle to be adapted, and then some shift, unexpected or otherwise, will provide the additional incentive, and suddenly we will experience yet another advance.

Author Paul Harmon

Executive Editor and Founder, Business Process Trends In addition to his role as Executive Editor and Founder of Business Process Trends, Paul Harmon is Chief Consultant and Founder of BPTrends Associates, a professional services company providing educational and consulting services to managers interested in understanding and implementing business process change. Paul is a noted consultant, author and analyst concerned with applying new technologies to real-world business problems. He is the author of *Business Process Change: A Manager's Guide to Improving, Redesigning, and Automating Processes* (2003). He has previously co-authored *Developing E-business Systems and Architectures* (2001), *Understanding UML* (1998), and *Intelligent Software Systems Development* (1993). Mr. Harmon has served as a senior consultant and head of Cutter Consortium's Distributed Architecture practice. Between 1985 and 2000 Mr. Harmon wrote Cutter newsletters, including *Expert Systems Strategies, CASE Strategies, and Component Development Strategies*. Paul has worked on major process redesign projects with Bank of America, Wells Fargo, Security Pacific, Prudential, and Citibank, among others. He is a member of ISPI and a Certified Performance Technologist. Paul is a widely respected keynote speaker and has developed and delivered workshops and seminars on a wide variety of topics to conferences and major corporations throughout the world. Paul lives in Las Vegas. Paul can be reached at pharmon@bptrends.com FacebookTwitter